Evolution of Cellular Systems
Optimization of Gene Regulatory Networks for the Development of Morphology and Control

Vom Fachbereich
Elektrotechnik und Informationstechnik
der Technischen Universität Darmstadt
zur Erlangung des akademischen Grades
eines Doktor-Ingenieurs (Dr.-Ing.)
genehmigte Dissertation

von

Dipl.-Ing. Lisa Schramm

geboren am 23. März 1983
in Frankfurt am Main, Deutschland

Referent:	Prof. Dr.-Ing. J. Adamy
Korreferent:	Prof. Dr. rer. nat. B. Sendhoff
Tag der Einreichung:	16. Juni 2011
Tag der mündlichen Prüfung:	28. Oktober 2011

D17
Darmstadt 2011

Acknowledgments

This work was conducted while I was working as scientific assistant at the control theory and robotics lab at the Technische Universität Darmstadt. Financial and conceptual support and computational power was provided by a cooperation between the Technische Universität Darmstadt and the Honda Research Institute Europe GmbH (HRI-EU). I want to thank many people for their support and encouragement which was invaluable for the successful completion of this PhD thesis.

Special thanks go to Prof. Dr. Adamy who encouraged me very early to start the PhD. He always believed in me and the success of my work and gave me the opportunity to develop myself.

The thesis would not have been possible unless the visions and numerous discussions with my advisor from the HRI-EU, Prof. Dr. Bernhard Sendhoff, thank you Bernhard. I am grateful to Prof. Dr. Yaochu Jin for many discussions, his guidance and that he was always pushing me to achieve more and Dr. Markus Olhofer for motivating me and working on the application-oriented view of my work.

I want to thank all colleagues from the control theory and robotics lab for the great atmosphere, the inspiring discussions in the lunch and coffee breaks and scientific discussions from different points of view. Thanks to my colleagues from the HRI-EU, especially those from the CSOA group that supported me with fruitful discussions, programming tips and much more.

Special thanks go to my former colleague Dr. Till Steiner. I have learned much from working with you, thank you for being a good friend and 'roommate'. I want to thank Dr. Ben Jones for many discussions and joining me at many conferences. I also want to thank Nils Einecke, Dr. Sven Rebhan, Dr. Jens Schmüdderich, Dr. Sven Schrader, and Dr. Daniel Weiler for their scientific (and non-scientific) discussions and also Dr. Jochen Eppler and Dr. Marc-Oliver Gewaltig for their support using NEST.

I want to acknowledge the debt I owe Dr. Giles Endicott, Sandra Hildebrandt, Dr. Ben Jones and Heiko Koch for proofreading and improving this thesis. I am grateful to Valentina Ansel, Sylvia Gelman, Birgit Heid, Dr. David Luttropp, Nevriye Memet, Andrea Voit, and Burkhard Zittel for their organizational and administrative support.

Special thanks go to my family and my friends for their assistance and support. Last but not least I thank my husband Heiko Koch for his incessant comprehension and support, for encouraging me and for tolerating my mood.

Contents

List of Abbreviations and Symbols

CPG	central pattern generator
DNA	deoxyribonucleic acid
EA	evolutionary algorithm
ES	evolution strategy
FP	functional proximity
GRN	gene regulatory network
NN	neural network
RU	regulatory unit
SNN	spiking neural network
SU	structural unit
$\mathrm{SU}^{\mathrm{die}}$	if this structural unit is active, the cell dies
$\mathrm{SU}^{\mathrm{div}}$	if this structural unit is active, the cell divides
$\mathrm{SU}^{\mathrm{move}}$	if this structural unit is active, the cell moves
$\mathrm{SU}^{\mathrm{neuron}}$	if this structural unit is active, the cell becomes a neuron
$\mathrm{SU}^{\mathrm{TF}}$	if this structural unit is active, a transcription factor is produced
TF	transcription factor

f	fitness value
$f(\alpha_1, ..., \alpha_N)$	fitness function depending on α_1, ..., α_N
μ	number of parents
λ	number of offspring
σ	strategy parameter (mutation rate)
p_{dup}	probability for gene duplication
p_{trans}	probability for gene transposition
p_{del}	probability for gene deletion
x	x-coordinate of 2-dimensional computation area
y	y-coordinate of 2-dimensional computation area

Abstract

This thesis focuses on the evolution of the development of simulated organisms, where a biologically inspired model to simulate the development of organisms is presented. The individuals are made up of several cells that mechanically interact with each other and can perform different actions, e.g. cell division and cell death. The development of the individual starts with a single cell, whose actions are controlled by a gene regulatory network. The motivation of this thesis is therefore twofold. One aim is to improve the computational experiments that can in the future shed light on transitions in evolutionary biology which can not be analyzed with the available biological data. The other aim is to use this knowledge to enhance the performance of engineering design, e.g. to optimize topologies.

In the first part of this thesis three different analyzes of morphological development are performed. A stable development that includes self-repairing behavior is evolved and analyzed. Dynamic stability can be achieved without fixing the cells on a grid or using contact inhibition, as most other models do. It is shown that genetic redundancy during the evolution is important and thus removing redundancy during the evolution decreases its performance. A new measurement for the probability of a redundant gene to become functional, the functional proximity, is presented. Genomes with a high functional proximity are shown to improve the evolution.

Most models in literature either optimize only the shape or the structure, or if both are optimized separated genomes are used. Evolving both concurrently and in one genome can result in positive effects on the resulting organisms. Such a model is presented in the second part. The individuals are evolved to fulfill a function that depends on shape and control. First, a spiking neural network is evolved, then a swimming organism is developed with a genome separated to describe the shape and control individually. In the final part, both parts, the development of shape and control are merged into one genome. Extending the concept of concurrent shape and control optimization using one genome to an engineering context may hold the potential for improvements on current practice.

Kurzfassung

Diese Dissertation befasst sich mit der Evolution der Entwicklung von simulierten Individuen. Dafür wird ein biologisch inspiriertes Modell entworfen, das die Entwicklung von Individuen simuliert. Die Individuen bestehen aus mehreren Zellen, die mechanisch miteinander interagieren können. Der Entwicklungsprozess der Individuen startet mit einer Zelle, diese kann sich teilen, sterben und einige weitere „Aktionen“ ausführen. Diese Aktionen werden von einem genregulatorischen Netz kontrolliert. Diese Arbeit hat zwei Hauptziele, das Modell soll die Möglichkeit bieten, einerseits das Verständnis von biologischen Prozessen zu erhöhen und andererseits technische Optimierungen zu verbessern.

Der erste Teil dieser Arbeit behandelt verschiedenen Analysen der Entwicklung von Formen. Als erstes wird ein stabiler Entwicklungsprozess evolviert. Dabei wird eine dynamische Stabilität erreicht, die sich von den meisten anderen Modellen in den folgenden zwei Punkten unterscheidet: Einerseits sind die Zellen nicht auf einem Gitter fixiert und andererseits wird die Zellteilung nicht direkt von der Anzahl der Zellen in der Umgebung beeinflusst. Zweitens werden Netzwerkmotive in den evolvierten genregulatorischen Netzen analysiert. Drittens wird gezeigt, dass das Entfernen genetischer Redundanz während der Evolution das Optimierungsergebnis verschlechtert. Verschiedene Messgrößen für Redundanz werden definiert und es wird gezeigt, dass redundante Gene, die sich häufig in nicht redundante Gene mutieren lassen, hilfreich für die Evolution sind.

Im zweiten Teil der Arbeit werden verschiedene Funktionalitäten der Individuen evolviert, zum Beispiel die Evolution schwimmender Individuen oder neuronaler Netze, um das Futterfangen einer *Hydra* zu simulieren. Bei den schwimmenden Individuen werden Form und Bewegung der Individuen gleichzeitig evolviert und verschiedene Strategien zum Schwimmen werden gefunden. Am Ende der Arbeit wird ein Modell entwickelt, bei dem die Form und Funktion der Individuen auf einmal und mit Hilfe nur eines Genoms evolviert werden kann. Die Möglichkeit, in der technischen Optimierung Form und Kontrolle gleichzeitig und in einem Genom zu optimieren, bietet ein großes Verbesserungspotential.

1 Introduction

Biology comprises of a wonderful variety of lifeforms by creating building plans that encode everything from relatively simple organisms up to extremely complex ones. The language of these building plans has not changed dramatically at least not on the same level as the transition from simple to complex phenotypes might initially suggest. The unifying theory of biological science is the theory of evolution [46]. Most adult multicellular organisms start from a single cell and grow to form complete organisms. This process, the development, includes cellular, genetic and environmental interactions.

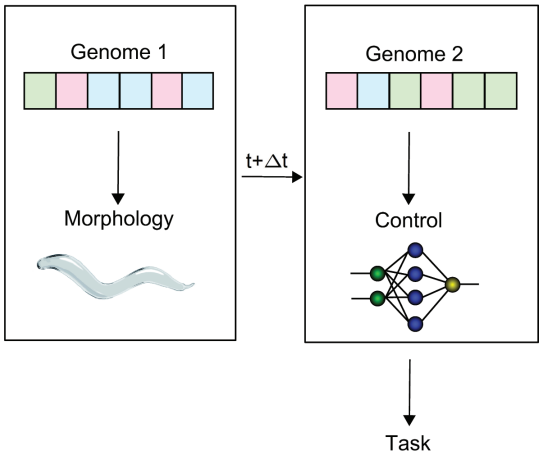
Evolutionary changes require several generations, so for most species they are difficult to analyze. The relationships between the different species are sometimes difficult to define, because it is very time-consuming and expensive to decipher the genome of a species. In particular, only a small amount of data from early evolution is available because most creatures from these periods are extinct. Drawing the phylogenetic tree, which shows the evolutionary relationships between species in a tree structure, is still a challenging task in biology. Simulating biological evolution in a computer can help to improve the understanding of biological data.

Using computational models and simulations to improve our understanding of biological systems has become a widely accepted approach in systems biology [78] and is sometimes referred to as *in silico* experiments (as opposed to *in vitro* and *in vivo*). Due to the enormous complexity of biological systems and the coupling of many different spatio-temporal scales, the level of detail that is required in simulations to make verifiable predictions and to provide meaningful analysis is often difficult to both specify and realize. One such time scale which plays an ubiquitously important role in biology is the evolutionary development of organisms. To date the study of evolution in the laboratory has been confined to viruses and bacteria. Therefore, to improve our understanding of the evolutionary influence on the structure and function of biological systems, we have to resort to computational experiments. Furthermore, as a result of the

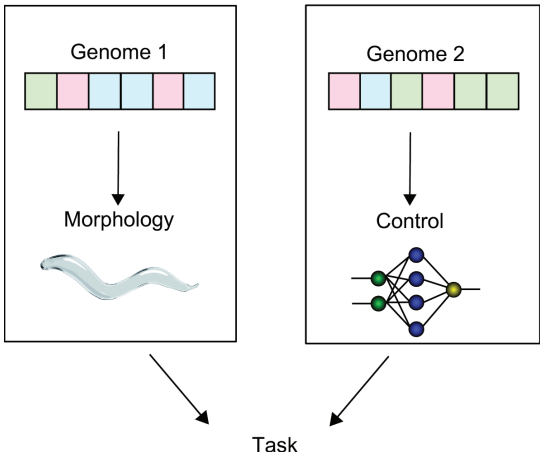
considerable increase of available computing power and refined simulation technologies, we are now in a position to perform simulations that are more closely aligned with biological systems.

In addition to reaching a conceptual understanding of biological evolution these concepts can also be used for engineering optimization. The optimization of shapes, structures or control systems are still challenging tasks. The idea of evolution has been transferred to engineering optimization problems by forming evolutionary algorithms, which are currently well established optimization techniques. The number of parameters to be optimized or the dimensionality of the search space increases with the complexity of the problem in direct encoding methods for evolutionary algorithms. This leads to the curse of dimensionality that can be solved with indirect encoding methods, e.g. gene regulatory networks. Indirect encoding methods use a non-trivial mapping from the genome to the resulting individual (genotype-phenotype mapping) and therefore there is no direct correlation between the dimensionality of the search space and the size of the chromosome. The developmental encoding is a biologically inspired indirect encoding method, where building rules are used, which start with a single unit (e.g. a cell) to build a complete system (the individual or phenotype). These building rules themselves can also be inspired by biology e.g. by simulating gene regulatory networks, which imitate the behavior of biological genes.

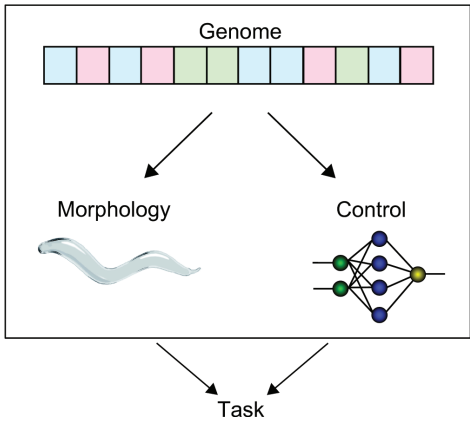
In those developmental models often only the shape or the control of a system is optimized. In current engineering design practices, e.g. to build robots, often the shape of the system is defined and subsequently its control is optimized. For these approaches, if the shape is changed, it is only changed after optimizing the control (see Figure 1.1a). The next step is to use approaches that optimize the shape and control concurrently with a separated genome for each, see Figure 1.1b. Robotics is one example where both, shape and control of a system need to be optimized. Using such an approach has the advantage that improvements in the shape can simplify the control and therefore improve the overall performance of the system. E.g. a passive dynamic walker can walk down a slope without control only because of its mechanical configuration [94]. Bongard analyzed robots that can change their morphology from anguilliform (eel-like) into legged during their lifetime and compares them to robots that do not change their shape [21]. He found robots that change their morphology in early stages of the evolution and gradually lose the anguilliform body plan



(a) First, the morphology is optimized, afterwards the control.



(b) Optimizing morphology and control in two separated genomes concurrently.



(c) One genome for morphology and control together.

Figure 1.1: Different possibilities for optimizing shape and control.

evolve faster and are more robust than robots that never change their shape. Therefore, the effects of coevolution of morphology and control e.g. the ones described by Bongard can improve the optimization results further. The approach presented here should define the morphology and control not only concurrently, but also in one genome, see Figure 1.1c.

The motivation of this thesis is therefore twofold. One aim is to contribute towards an improvement of computational experiments that can in the future shed light on transitions in evolutionary biology which can not be analyzed with the available biological data. The other aim is to use this knowledge to enhance the performance of engineering design, e.g. to opti-

mize topologies. The resulting structures of such a model are often more complex than those designed by an engineer and therefore more difficult to build. Complex 3D structures can be produced by e.g. a 3D printer in a process termed rapid prototyping. These printers can also be used for manufacturing on larger scales and are likely to become cheaper in the future.

Therefore, I outline a simulation environment that evolves a model of a gene regulatory network (GRN) combined with a cellular model with physical interactions. Using such a developmental model has the potential to provide better scalability than a direct encoding, because the number of parameters is not directly coupled to the complexity of the problem.

On the one hand, the understanding of early evolution and especially its principles needs to be improved. The formation of the first control structures and how early coevolution between morphology and control emerged is one question. Investigating brain-body coevolution with such biologically plausible models should gain deeper insights into the coevolution of nervous systems and morphologies in biology. The model designed here uses a biologically inspired encoding and should therefore provide the opportunity to produce results that improve the understanding of biological evolution, but a direct comparison to biology is not an aim of this thesis.

On the other hand, it should be possible for the proposed system to be used in engineering optimization e.g. to build robots or optimize the structure of a material. A major advantage of such models is that shape and control can be optimized concurrently, as described in Figure 1.1. Evolving morphology and control together in one genome, which is different to most approaches in the literature is a major aim of this thesis. The possibility to evolve morphology and control together provides the opportunity that their complexity increases concurrently and therefore it could improve the optimization process. The tasks for the evolution in this thesis are examples, e.g. a neural network that controls food catching or performs movements for swimming. The individuals, which are artificial animals, are termed animats. In this thesis only simulations of the individuals are used, as building these robots in real hardware is beyond the scope of this thesis due to the additional complexity involved.

1.1 Overview of the Thesis

This thesis starts with a biological overview, first about development, especially some details about the two organisms *Hydra* and *C. elegans* which

are used as examples in this thesis. Secondly, an introduction to gene regulatory networks and thirdly, network motifs in gene regulatory networks (GRNs). This is followed in Chapter 3 by an introduction to technical applications and starts with evolutionary strategies and a literature overview of their encoding methods and the simulation of artificial individuals. In Chapter 4, I propose the basic model that is developed in this thesis. This is separated into the evolutionary strategy, the gene regulatory network model and the model of the mechanical interactions of the cells.

The thesis is subdivided into two parts, the first part deals with the evolution of morphologies and starts with a basic experiment which is used in the following three chapters. It contains the evolution of an elongated morphology, which is inspired by the topology of *C. elegans*. The first analysis of the development of such organisms is about the mechanisms of stability and regeneration. In Chapter 7, different network motifs are counted and their changes on an evolutionary scale are analyzed. Furthermore, I analyze redundancy in the genome from an evolutionary perspective.

The second part deals with the evolution of morphology and control, contrary to the first part that only analyzed morphological development. Different animats (artificial animals) are simulated. In Chapter 9 the food catching behavior of a *Hydra*-like animat is evolved. To achieve this, a spiking neural network is developed including a developmental order similar to the biological organism. A morphology with more functionality is evolved in Chapter 10. The individuals should swim in a simulated environment, whereas a simple encoding of the movements is used. Since the evolution of the spiking neural network itself is very complex, I used a less complex neural network to evolve the topology of *C. elegans* with a swimming function, both controlled by the same chromosome. This thesis is concluded with a summary and an outlook. An overview of the different encoding concepts of the chapters can be found in Figure 1.2.

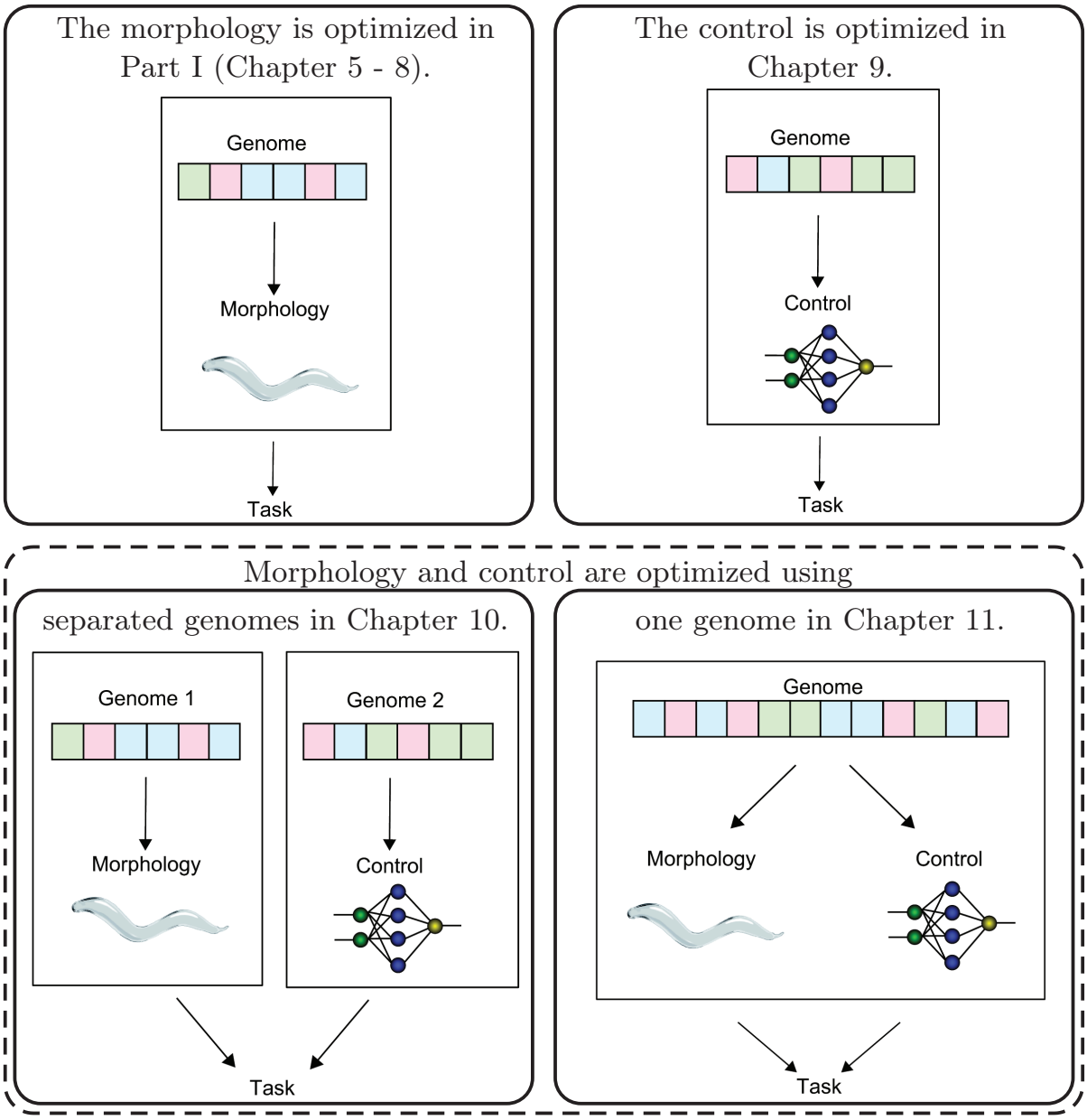


Figure 1.2: Conceptual overview of this thesis.

1.2 Contributions

The major contribution of this thesis is the design of a model for the development of multi-cellular organisms that can be used for evolutionary optimization. The contributions in more detail are twofold and listed in the following:

- The first part deals with analyzes of the developmental and evolutionary processes for shape formation.
  - The achievement of stability and regeneration during the development.
  - The occurrence of network motifs and their changes during evolution.
  - The role of redundancy in the evolution of gene regulatory networks and their influence on the performance of the optimization.
- The second part handles functional tasks and the ability to evolve them.
  - The evolution of a spiking neural network to catch food with developmental rules.
  - The evolution of the development of a morphology with a complex function (swimming) and a simple encoding for the control part.
  - The evolution of the development of a morphology with a complex function (swimming) and central pattern generators for the control part, where the morphology and the control is defined in one genome.

These contributions are discussed in more detail in Chapter 12.

2 Biological Background

Evolutionary theory deals with the diversification and speciation of life on earth. Evolutionary changes are observable in the timescale of generations, which can last between hours and decades. Each individual itself undergoes a developmental process in which the individual grows depending on its genes and environmental influences.

With his seminal book *On the Origin of Species* Charles Darwin uses natural selection as an explanation for the diversification and speciation of species [30]. The well known *survival of the fittest* was added in the 6th edition of Darwins *On the Origin of Species*, proposed by Herbert Spencer.

The current theory of evolution is based on the competition of species (Red Queen model) and the response to changes in the environment (Court Jester model), which take place on different time scales [15]. The theory generally describes the cycle of reproduction and selection of individuals and the resulting changes in the populations. For a detailed description of the theory of evolution see e.g. [30, 46, 48].

All individuals are based on their genetic information, termed the *genotype*, which is encoded in deoxyribonucleic acid (DNA) as found in all cell nuclei. The *phenotype*, which is the individual itself, is built or grown from the zygote[1)] following rules encoded in the DNA. The DNA and the environment, i.e. both influence the development of organisms phenotypes (see Section 2.1). Individuals that reproduce often are named fit individuals. The fitter they are, which means the more often they reproduce, the higher the probability that their genetic information is passed on to later generations. Therefore 'good' genetic material is passed on more often and as a restult establishes itself.

The next section introduces biological development where two example organisms are presented. This part is followed by a highly simplified description of biological genes and their interactions. The following chapter presents applications of evolution and development.

[1)] The zygote is the first cell of an organism and is produced by fertilization.

2.1 Early Biological Development

Nature discovered an astonishing ability to grow complex organisms starting from one single cell. Although development differs significantly between individuals, some similarities can be found. During early development, generally fast cell divisions (cleavage) without enlargement of the overall organism takes place. In some individuals the cells reorganize themselves in the gastrulation process [51]. The concentrations and gradients of some chemicals determine the early development and define the axes of the organism. The first gradients are provided by the mother individual (maternal gradients). As axis formation starts, the axes can become anterior-posterior (head to tail or mouth to anus), dorsal-ventral (back to front) or right-left and provide positional information for the cells to specialize [51]. Circular and bilateral axes exist, e.g. the *Hydra* which is described in the next section has radially symmetric axes. The *C. elegans* (see Section 2.1.2) is an example of a bilaterally symmetric organism. Most higher developed organisms are also bilaterally symmetric.

The DNA which can be represented and analyzed as a gene regulatory network, consists of genes, see Section 2.2 and [2]. Both, the genes and the internal cellular environment influence the behavior of the cells. Influences can be chemical or physical, e.g. chemical concentrations, pressure, adhesion or repulsion forces.

The next two sections describe two organisms which are both primitive in evolutionary terms, the *Hydra* and *C.elegans*. Both are well studied in the field of biology and are especially well received in the study of primitive nervous system evolution.

2.1.1 The Cnidarian Hydra

The freshwater hydrozoa (*Hydra*) is most widely known due to two remarkable properties: 1) It is the first organism having a nervous system - biologists speak of nerve nets because of its seemingly random connectivity pattern (although this has been challenged in the more recent literature [83, 99]) and 2) its astonishing regeneration capability [59]. A schematic drawing of the body morphology is shown in Figure 2.1.

The freshwater hydrozoa (*Hydra*) is about $0.5 - 2$ *cm* long and lives in fresh water. Biologically it belongs to the phylum Cnidaria and like many jellyfish species it is radially symmetric. Its shape is essentially a tube with tentacle extensions at the top (see Figure 2.1). The body wall consists of an inner endoderm, the mesoglea and an outer ectoderm. The endoderm

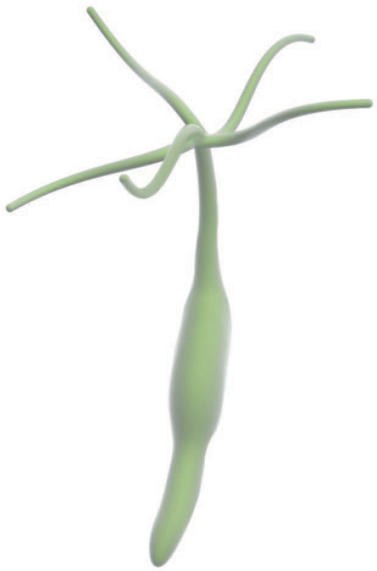


Figure 2.1: Schematic drawing of the body of *Hydra*.

and the ectoderm form an epithelial layer. The neurons of the *Hydra* reside within the mesoglea. Its tentacles are used to catch food, relying on an olfactory sensor to detect prey [27]. Movements of the tentacles and of the body are controlled by a nerve net.

For biologists, the *Hydra* is an interesting organism because of its amazing ability to regenerate. If it is cut into several pieces, a small 'head' and a small 'foot' form at each piece. This happens with little cell division; the small regenerating *Hydras* grow through a reorganization of the existing tissue pieces in a process known as *morphallaxis* [51]. All cells including those of its nervous system regrow during its whole lifetime [23, 71]. With enough food, the forming *Hydras* eventually reach normal size. Even if the pieces are put into a centrifuge, they can actually reaggregate into a full *Hydra* [82].

The regenerative capacity of *Hydra* is strongly linked to its continuous morphogenetic plasticity, i.e. the constant cell production and tissue turnover, and to the embryogenesis of cnidarians, which however is not yet fully understood [18, 19]. According to Holstein et al. there are homologous genes that are involved in bilaterian embryogenesis and *Hydra* regeneration [59]. In *Hydra* regeneration both *morphallaxis*, i.e., the redifferentiation or reprogramming of cells, and *epimorphosis*, i.e., the proliferation of new cells, seem to play a role in regeneration. Due to the

Figure 2.2: Schematic drawing of the body of *C. elegans*.

high-level of morphogenetic plasticity, it seems that both cell proliferation and cellular differentiation is always possible in *Hydra*. However, in any case it is important that some position dependent signal that can guide the regenerating tissue is available during the regeneration process. In a first step these patterning signals have to be locally self-generated as a response to lesions and in a second step they have to be used by the regulatory system to trigger cell proliferation and differentiation.

2.1.2 The Nematode C. elegans

Caenorhabditis elegans (*C. elegans*) is a small nematode that lives in the soil [2]. The body plan of *C. elegans* is bilaterally symmetric, elongated and approximately 1.2 *mm* long. A schematic drawing can be found in Figure 2.2. Its development is well studied and reproducible, which means it is the same for all *C. elegans* worms. Because of this and because it is very robust and has a simple metabolism, the *C. elegans* is a good model organism. Its genome is completely sequenced [58]. The adult *C. elegans* consists of exactly 959 body cells, where 302 are neurons and 56 are glial cells[2)] [57]. The positions of all cells are identified and the neuronal connectivity is known.

C. elegans can swim in water or crawl on and through soil. These movements produce two different patterns, which are both undulatory. Pierce-Shimomura et al. recorded the different patterns of movement of *C. elegans* [107]. They observed different frequencies (peak at ≈ 0.8 Hz for crawl and at ≈ 2.1 Hz for swim in power spectrum analysis of its bending). They show that the swimming is not a faster version of crawling, so there is a qualitative difference in frequency, amplitude, and the propagation of the dorsal-ventral bends. Berri et al. showed a continuous transition between swimming and crawling while changing the density of the environment in small steps, which leads to different drag coefficients [16].

[2)] Glial cells are in general no neurons but support them e.g. they guide neuron migration and and axon growth during development.

2.2 Biological Gene Regulatory Networks (GRNs)

The development of organisms is controlled by their genes and their environment. The proteins that influence the majority of the chemical processes in the cells are built using the genes as a blueprint. The mechanisms of cellular activities, mainly the process of building proteins, are described in the following, starting with an extremely coarse and general explanation, which is followed by a more detailed description. A thorough description can be found in e.g. [2, 51].

All cells of an individual have the same genome, the sequences of the genome are termed genes. Each gene can influence the behavior of the cell and therefore control its growth, death, differentiation, the chemicals the cell produces and many more 'actions' of the cell. The execution of the genes can be activated or repressed by the interplay of many chemicals. Since genes can influence the production of chemicals they can influence the activation or inhibition of themselves or other genes. For the activation of a gene, several chemicals are needed for the reaction. The concentration of the chemical influences the probability for this reaction. A chemical produced by e.g. gene A can influence (positively or negatively) the activation of e.g. gene B. These highly nonlinear interactions between the different genes can be described in a gene regulatory network (GRN). For visualization, genes can be plotted as dots, and activation or repression between those can be drawn as arrows. Chemicals can also be exchanged between the different cells, so the neighboring cells can also influence the activation of genes and therefore the behavior of a cell.

In biology, the cells of all eukaryotes[3)] have chromosomes that contain their genetic material, the DNA. The DNA itself has a double helix structure; each strand consists of many nucleotides which are made of a sugar-phosphate molecule combined with a base [2]. The four bases are adenine (A), guanine (G), cytosine (C), and thymine (T). Two strands form the double helix, where two complementary bases are connected together (always A with T and C with G).

The process of transcription inside the nucleus builds ribonucleic acid (RNA) which consists of nucleotides with the bases A, G and C but instead

[3)] Eukaryotes e.g. animals, plants and fungi are organisms with cell nuclei, whereas the cells of prokaryotes e.g. bacteria have no nucleus. The nucleus contains the DNA and is surrounded by cytoplasm.

of thymine it consists of uracil (U). The complementary base pairs of the segments of DNA (genes) are used to build the RNA. Proteins are built in the cytoplasm using the RNA as blueprint during translation. There exist a lot of differences between the prokaryotic and eukaryotic transcription and translation of genes. A simplified process is described here. The whole complex process is still not fully understood.

There are different types of RNA that are produced by different genes. The RNA that will become messenger RNA (mRNA) is called pre-mRNA. The eukaryotic DNA and therefore the RNA consists of exons (expressed sequences) and introns (intervening sequences). The introns are removed from the pre-mRNA in a process of splicing, which results in mRNA. Again with a complex chemical process some introns are removed from the pre-mRNA. Different mRNAs can be transcribed from one gene, because the pre-mRNA can be spliced at different positions. There exist other types of RNA, called non-coding RNA that are also needed for different processes.

During the transcription process, the RNA polymerase[4)] binds to the DNA. The RNA polymerase is a molecule and moves along the DNA to produce mRNA. Therefore, it unwinds the DNA helix and adds nucleotides to the RNA. The nucleotides are added using the complementary base pairs to one DNA strand (G-C and A-U). After its creation, the mRNA strand separates from the DNA and the DNA double helix closes again.

The RNA polymerase needs a promoter region on the DNA to start and a terminator region to stop the transcription. The promotor and terminator are both defined patterns on the DNA. Transcription factors (TFs), that are also proteins, are attached to the promoters by the RNA polymerase. In eukaryotes, the RNA polymerase additionally needs many other proteins to start the transcription process. Different general transcription factors bind to the promoter of the DNA and provide the opportunity for the RNA polymerase to assemble at the promotor. Additionally to the general TFs, transcriptional activators need to dock to enhancer regions, that can be at other positions on the DNA. Some other proteins together form a mediator, that bounds the activator to the RNA polymerase. Generally, many proteins are needed for the transcription process (over hundred units).

In eukaryotes, transcription is completely inside the nucleus of the cell, some types of RNA e.g. the mRNA need to move to the cytoplasm. Nu-

[4)] There are three different types of RNA polymerase in eukaryotes that transcribe different types of genes.

clear pore complexes (channels) transport the RNA to the cytoplasm. They transport only sound and useful RNA, that is marked by proteins.

The information encoded in the mRNA is used to build proteins in a process called translation. Ribosomes uses tRNA (another type of non-coding RNA) to build proteins encoded in mRNA. The tRNA consists amongst others of an amino acid and some bases. The ribosome itself is transformed rRNA and some more proteins. Three bases (A, G, C, U) on the mRNA are termed a codon. Three of the bases on the tRNA build an anticodon. The complementary anticodon of the tRNA binds to the codon on the mRNA using the ribosome. The amino-acid of the tRNA is added to a chain until a stop codon is reached. The process begins at a start codon.

The resulting chain of amino-acids (a polypeptide) separates from the ribosome and folds into a protein, which has a unique 3D structure (conformation). The structure is defined by minimizing the free energy and can change slightly when the protein interacts with other molecules. Different levels of protein structure are defined in biology. The polypeptide is termed primary structure of a protein. This sequence folds into often occurring motives, its secondary structure. The complete 3D structure of a protein is called its tertiary structure. Proteins can assemble to a complex of more than one polypeptide, which is the quaternary structure, e.g. hemoglobin.

The whole interactions from the genes to the proteins and their feedback to the activations of the genes are described in gene regulatory networks (GRNs). GRNs can be visualized as directed graphs, where the vertices represent the genes and the edges represent the activational relation.

To sum up, the expression of genes consists of the following steps and can be regulated in all steps. (taken from [2]):

- Transcriptional control (whose parts of the DNA are transcribed to RNA)
- RNA processing control (e.g. splicing)
- RNA transport and localization control (which RNA and where it is transported out of the nucleus)
- Translation control (which mRNAs are translated to proteins)

- mRNA degradation control and RNA interference (RNA can be destabilized (not described here))
- Protein activity control (proteins can selectively be activated or deactivated (not described here))

The proteins in general can influence the execution in all steps. Because of the ability of the proteins to influence their own production and degradation, there is a strong feedback. There is also the dependency on the cells environment, because proteins can also be sensed by or transported into cells. Especially in the early development of an organism, as described in Chapter 2.1, the maternal gradients strongly influence the behavior of the cells.

The amount of genetic data that has been deciphered is growing, but a considerable amount has not yet been analyzed and the analysis is still time consuming and expensive. Computational approaches can therefore help to simulate, predict and understand the genetic data. Some techniques to decipher genetic information are described in [51]. Deciphering the genetic data is only the first step, an understanding of the complex interactions and their influence on the developmental process is also crucial but still in its infancy [47]. Besides taking direct measurements, there are other approaches to gain more knowledge about biological systems. Geard and Willadsen provide a good overview of different methods to predict, model and simulate the structures of GRNs [47], while Samsonova et al. present an approach to predict the tissue-specific gene expression using a supervised learning framework [111].

A lot of approaches measure or predict some connections between some genes of one organism, some are presented in the following. They often analyze only few connections or one type of connection, that also confirms its difficulty and the high expenses.

Ashe and Briscoe [6] review the role of morphogen gradients in early development. They outline different signalling pathways and some strategies for the regulation of genes. The number of responses a morphogen gradient can generate, which is equal to the number of different concentrations which cause different behavior of cells, were analyzed empirically. Between three and seven distinct thresholds were identified. Cooper et al. [28] predict the types of different feed forword loops in *E. coli*.

Quayle and Bullock model the evolution of gene regulatory networks and analyze the occurrence of different cyclic patterns [109]. The distribution of their artificial genetic networks lies between those of random

networks and scale-free networks[5]. They evolve cyclic patterns with different lengths and show that the system can evolve short patterns but the performance decreases rapidly for longer patterns. It is not only the length of the pattern that constrains the performance of the evolution, also some pattern lengths seem to be more difficult. This suggests that evolution generates complicated expressions from building blocks of shorter patterns.

Simulations of the generation of regulatory networks using duplication and mutation also show scale-free and small world topologies[6] [87]. Their distributions are comparable to those of *E. coli* and *S. cerevisiae* and differ from random networks.

Pfeiffer et al. simulate metabolic networks based on a setting proposed in [72]. They show the emergence of hubs and a connectivity distribution that is comparable to natural networks of a similar size [106]. Additionally, they predict that group transfer plays a key role in the emergence of hubs. Ciliberti et al. analyze the robustness and its gradual evolution in GRNs [26].

2.3 Network Motifs in Gene Regulatory Networks

According to a review by Babu et al. the analysis of gene regulatory networks can be divided into three different approaches: firstly, the connections between transcription factors and target genes, thus the activations of the different genes [8]; secondly, the network motifs, patterns in the GRN that are over-represented. The review by Alon identifies network motifs that are found in bacteria, e.g. *E. coli*, yeast and some other plants and animals [4]. Thirdly, larger modules and connectivity hubs.

To analyze large transcription networks, block patterns can be built and network motifs can be defined. Alon defines network motifs as patterns, that occur in real networks significantly more often than in random ones, assuming that the random networks have the same characteristics as the real network (e.g. number of nodes, edges, etc.) [3, 4]. Some motifs are discussed in Chapter 7 in more detail.

[5] Scale-free networks have a degree-distribution which can be modeled with a power law function i.e. the probability that a node selected uniformly at random has a certain number of links (degree), follows a power law function [11].

[6] Small-world networks are characterized by a short average path length between two random nodes [138]. Scale-free networks can exhibit the small-world property.

Analysis of biological data revealed that such motifs can widely be identified in bacteria and yeast, see [8]. Most recently, it has been found that some motifs may have played an essential role in evolution. For instance, Kwon and Cho analyzed the role of feedback loops and found that more positive feedback loops and less negative feedback loops contribute to the robustness of the regulatory system [88].

An analysis of network motifs in the model presented in this thesis is described in Chapter 7.

3 Artificial Evolution and Development

This chapter contains an overview of artificial evolution strategies and their encoding methods. This is followed by an overview of the related work on artificial developmental models and gene regulatory network models. The chapter is concluded with related work on the simulation of animats.

Evolutionary Algorithms (EAs) belong to the huge group of probabilistic optimization algorithms. They work by imitating natural evolution and simulate the evolutionary cycle in terms of reproduction, recombination, mutation and natural selection (see Figure 3.1 and e.g. [7, 9]).

At the beginning of an artificial evolutionary cycle μ parents are initialized. The initialization can be random or can contain previous knowledge about the problem to be solved. The chromosomes can be real valued, binary or based on any other suitable encoding alphabet. Each contains a candidate objective parameter set of some function that is to be optimized. Parent chromosomes are recombined and mutated to build λ offspring. For the recombination process, two or more parent individuals (chromosomes) are chosen and parts of each are combined to build a new one. Mutation adapts some of the objective values in the chromosome depending on a

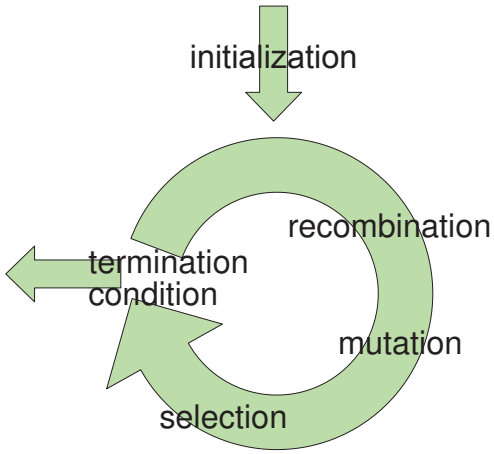


Figure 3.1: Loop of a standard evolutionary algorithm.

strategy parameter σ. When the offspring chromosomes are built, their fitness, the outcome of the function that should be optimized using the given chromosome, is computed. Depending on fitness μ parents are selected. When using elitism, the parents of the next generation are chosen from both the parents and the offspring of the current generation, while without elitism they are chosen only from the offspring. Thereby, elitism ensures that the fittest individual always passes to the next generation. The evolutionary cycle is iterated until the fitness is deemed good enough or after a fixed number of generations [17]. The optimization can be a minimization or maximization task and with this a good fitness can have high or low values. In this thesis all optimizations are minimizations and therefore low fitness values represent good individuals.

The chromosome can be encoded in a variety of ways. A direct encoding explicitly uses the values of the chromosome, e.g. as coordinates of points, as connection strengths of a network or as parameters of a mathematical function. In a direct encoding and with increasing complexity of the problem, e.g. an increasing number of dimensions, the size of the chromosomes also increases which leads to the curse of dimensionality [75]. Indirect encoding methods typically employ a grammar allowing the solution to be indirectly built up. An advantage of these indirect methods is high scalability.

Artificial development, often referred to as artificial embryogeny [123] or artificial ontogeny, simulates the natural developmental process using different levels of abstraction. Nature uses a relatively small genome which is interesting because of the higher number of cells in a multi-cellular organism. Each cell is itself highly complex, while millions of them form a complete body and a complex shape [53]. The developmental process of an animal starts with a fertilized egg, that grows into a mature multi-cellular organism as described in Section 2.1.

A central theme is to understand how compact codes can be used to build complex systems. It can be divided into two main questions, one is to understand biology and the other is to improve design processes. Since there is a large difference between these two, different levels of abstraction in the different models are chosen depending on the question of research.

To encode a neural network, it is possible to use an $n \times n$ connection matrix, where n is the number of neurons. There are several ways to encode this information, but for all direct methods, the chromosome length increases with an increasing neural network size. This is very problematic for large structures due to poor scalability. Indirect encoding methods

are typically rule-based and provide the possibility of re-using elements. Therefore the chromosome length does not depend directly on the number of neurons. Hence scalability is significantly improved.

If the genotype is smaller than the phenotype, the information is compressed and the genotype size is no longer directly correlated to the phenotype size. With some encodings, several phenotypes cannot possibly be constructed, therefore, parts of the search space can be excluded for the evolutionary algorithm [84]. This can not only hinder but also promote the evolutionary process if regions of the search space are smoother or denser. It is also possible to reuse elements, e.g. a leg of a robot may be described only once and reused. Developmental encoding is one indirect encoding method that can solve the problem of scalability, plasticity and robustness [53].

A direct encoding based on cellular automata[1)] outperformed the developmental model of Harding and Miller for most complexities [54], although the search space for this problem was relatively small. A comparison for larger search spaces would be interesting.

Bentley and Kumar also compared direct and indirect encodings [14]. In their comparison, an implicit embryogeny approach outperformed approaches with external and explicit embryogeny and without any embryogeny at all. It also scaled better.

Indirect encodings can be divided into two different types: grammar based approaches and cell chemistry approaches, but the distinction is often fuzzy. The grammar based approaches have a higher level of abstraction, the cell-morphogen approaches are more biologically motivated. Stanley and Miikkulainen develop a taxonomy for artificial embryogeny based on cell fate, targeting, heterochrony, canalization, and complexification [123]. Different approaches are categorized according to this taxonomy. Some of the various models are described in the following:

Grammatical encoding uses a recursive rewriting grammar [52, 75]. Kitano uses rules to transform 1×1 matrices into 2×2 matrices [75]. Then all elements of the new matrix are again transformed into 2×2 matrices and so on. Kitano shows, that the convergence of an evolutionary algorithm with grammatical encoding is faster than convergence with a direct encoding method [75], but Siddiqi and Lucas show that a direct encoding is as efficient as Kitano's grammatical encoding method [119].

[1)] Cellular automata use a grid of cells with discrete states and a set of rules to define or change the states depending on the states of the neighboring cells.

Cellular encoding (CE) uses a graph grammar method for encoding similarly structured neural networks [52, 103]. Gruau uses a grammar tree with ordered branches [52]. The nodes are labeled with the names of program symbols, which represent instructions for cell processes. Comparable to a Turing machine[2)], a cell executes its instructions step by step. Comparing cellular encoding and Kitano's grammatical encoding we can make the following observations [52]. First, representing repeated patterns is possible in both approaches. Second, Kitano's scheme requires an $m \times m$ matrix for a network of n neurons, with $m = 2^k$, $k \in N$ and $m \geq n$. Therefore, the resulting matrix is often larger than the number of neurons. For CE there is no similar condition. There, it is possible to produce 'infinite' networks by using loops within the grammar. The grammatical encoding method is limited by the maximum size of the connectivity matrix. Third, only some of the rules of Kitano's scheme are used. CE uses all rewriting rules, i.e., the trees can be more compact than Kitano's rules allow for.

Super coupled map (SCM) [76]: This second model proposed by Kitano is a continuous-valued, but discrete time model that is based on metabolic reactions. Rules depending on concentrations of several chemicals are encoded in the chromosomes.

Lindenmayer systems (L-systems) are a group of different grammatical encoding methods to model developmental processes [89]. Amongst other approaches, they are used for evolving virtual creatures or multicellular development [60]. It is a rewriting grammar that can either be context-free or context-sensitive and provides rules to modify strings that can represent multicellular creatures.

NeuroEvolution of Augmenting Topologies (NEAT) is a powerful direct method used to evolve neural networks (NNs) that efficiently optimizes their structure. NEAT uses three important methods: 1) genetic crossover operations are kept consistent through markings on the genes, 2) speciation allows individuals to evolve in a niche with a 'separate' structure optimization, 3) minimizing the dimensionality of the NN by starting with zero hidden nodes [122]. The **Hypercube-based NeuroEvolution of Augmenting Topologies (HyperNEAT**) is an extension of NEAT used to evolve connective compositional pattern producing networks. It is used to build large-scale NNs by exploiting the regularities along geometric dimensions [124]. In contrast to NEAT, HyperNEAT is an indirect approach.

[2)] The Turing machine was invented by Alan Turing and can simulate the logic of any computer algorithm [132, 133].

Komosiński and Rotaru-Varga compare three different encoding methods to evolve agents using the **Framsticks** system to maximize the height of the agents or their velocity [84]. The bodies of agents consist of so called *sticks* which have two connected endpoints and various physical properties. Two indirect encodings (direct recurrent and developmental) are found to outperform the direct encoding in all tasks.

Basanta et al. evolve static and dynamic stability using a model based on **cellular automata** [12]. Some of their individuals perform self-repair. Cells can divide, move or die. They have a fixed rule set of size 100 for the encoding of the genome. Rules depend on the overall run-time, the number of neighboring cells and their positions and the number of divisions the cell has already performed. Basic ingredients like cell position or the number of cellular divisions is prespecified in order to enable convergence to a stable state. Nevertheless, the results are interesting and the authors evolved systems that exhibited regeneration.

Fleischer and Barr used a **differential equation model** to hand-code simple development early in the field [43, 44]. They use a multi-cellular model with physical interactions to create artificial neural networks. They model diffusing chemicals for cell communication which change cell states and cell outputs as defined by differential equations.

Gene Regulatory Networks models use biological GRNs as example. They exist in different levels of abstraction and are described in the following section.

A few computational models for neural development are also reported. Cangelosi et al. [24] suggested a developmental model for cell division and migration that uses a rule-rewriting grammar. A model for neurogenesis that includes metabolic reactions and diffusion was proposed by Kitano [76], though no functionality of the developed neural network has been considered in the paper. A recurrent artificial neural network was used for modeling the development of a spiking neural network for the control of a Khepera robot [41].

A concise review of artificial development has recently been compiled by Harding and Banzhaf [53]. In particular, there are several approaches that simulate biological development based on gene regulatory networks and cell growth, see [29, 40, 66, 67, 125] and also the following section. Zhan et al. use gene regulatory networks to design electronic circuits [144].

A well-established problem is the evolution of a French flag [141]. The cells should differentiate to different types (which are represented in discrete colors or continuous grayscale) at different postions in the computa-

tion area. The resulting cellular pattern is then compared to a French or another flag. A general problem of optimization and also developmental evolution is to gain stability and the ability of regeneration. Miller developed the French flag and showed that some flags are stable. They do not change if development continues. Some also have the ability of repairing themselves if only small parts of the pattern are destroyed [100]. In the gene regulatory network model of Knabe et al. cells consist of several pixels and it is also used to solve the French flag problem [81].

Federici and Downing also evolved fault-tolerant developmental systems using a recursive neural network having different developmental stages [42]. During evolution, they progressively complexify the developmental process with chromosomes that are different at each stage of development.

3.1 Artificial Gene Regulatory Networks

A number of computational models have been developed to model biological gene regulatory networks [33], either for the re-construction of biological gene regulation subnetworks using biological data, or to simulate biological signal transduction or development. In the latter case, the aim is often to analyze fundamental properties such as robustness in systems biology, and for simulating important phenomena in artificial life.

Although GRNs mainly describe activations of different genes or nodes, the transitions between the nodes can be represented in many different ways, e.g. directed graphs, Bayesian networks, Boolean networks, ordinary and partial differential equations, qualitative differential equations, stochastic equations, rule based formalisms, all of which simulate the GRN on different levels of abstraction. See the reviews of de Jong and Hasty et al. for an overview of modelling biological GRNs to gain knowledge about biological organisms and principles [33, 55].

There are several types of GRN models. System states and variables can be logical or continuous, the model can be deterministic or stochastic, the update of the system states can be synchronous or asynchronous and a spatial structure can be incorporated or excluded [47]. The models can be divided into the following types (taken from Geard and Willadsen [47])

- In **Boolean Networks and Logical Models** genes can be either on or off depending on logical functions with the activations of the other genes and the network weights as variables. Random Boolean

Networks use a randomly generated network structure and a random selection of logical functions.

- **Scale-Free Networks** are Boolean networks with a structure comparable to small-world networks. This results in robust behavior and has a more biologically plausible structure.
- **Canalized Boolean Networks** use logical functions where one input can overrule all other inputs.
- **Logic Networks** use Boolean or multivalued logic functions which can simulate multiple thresholds and therefore provide a timing mechanism.
- In **Differential Equations Models** (see next section) gene products have continuous concentrations. They use linear or non-linear differential equations as update functions. The differential equation models are biologically more plausible than the Boolean networks, but contain many more parameters. They also assume that the gene expression and production is continuous, which is not always valid in biology.
- **Stochastic Models** simulate a production that is non-continuous. Stochastic models can be based either on Boolean or differential equations.

Models that combine some of the above features also exist.

Crombach and Hogeweg use a Boolean threshold network, with varying evolutionary goals to gain evolvability [29]. They find activations of genes that alternate very fast during evolution and name this evolutionary sensors.

3.2 Differential Equation (DE) Models for GRNs

There are several models that simulate GRNs using DEs [5, 38, 67, 86, 97, 125, 127, 129].

Mestl et al. analyze their model based on DE to find steady points [97]. This analysis is possible, because they use step functions as threshold functions, so the phase space can be easily divided. Boolean functions describe each threshold and the behavior can therefore be analyzed analytically.

The model proposed by Eggenberger is based on continuous-valued regulatory and structural units [38]. The parameters of a structural gene determine what happens and how it happens if the gene is activated. Activated structural units (SUs) can produce chemicals for cell-cell communication. Regulatory units (RUs) control the activation of the structural units. They use the concentration of the chemicals located in close proximity to the cell, as input. The RUs and SUs have an affinity parameter, used to determine which chemical substances they can interact with. The SUs can cause different cell behaviors, e.g. the production of chemicals for different signaling and controlling tasks, cell differentiation, cell division or cell death. Eggenberger uses cells in a 3D space which are fixed on a grid. The direction of the divisions are random, if no space is left, the cell does not divide. He shows how such a GRN using RUs and SUs can evolve simple shapes.

Eggenberger extends his model and evolves the GRN to develop a neural network that controls a foveating retina [40]. He also proposes the benefit of asymmetric cell division [39].

The model by Steiner et al. is based on the Eggenberger model. They evolve a developmental process resulting in an L-shape [125]. They extended their cell model to 3D to evolve stable and lightweight materials with an inner structure [127]. Two different cell types define the positions with and without material. This work is extended by the use of polarized cells and the evolution of complex maternal gradients [129].

Joachimczak and Wròbel also use a model based on the work of Eggenberger [67]. They evolve cells in a 3D space without relying on a grid to develop morphologies. The affinities of the RUs and SUs (which they name promoters and genes, respectively, moreover a gene is a regulatory unit in their terminology) are described by distances in an N-dimensional space. First results show no differences in the evolvability while changing the size N of the search space. They also use their gene regulatory model, to solve French flag type problems in 3D [66].

Andersen et al. use a GRN model based on the approach of Eggenberger and evolve stable development and which further shows a capacity for self-repair [5]. The cells are fixed on a grid, they use contact inhibition, so a cell surrounded by other cells is not able to divide.

Small subnetworks that perform basic tasks, e.g. bistable switches or oscillators are evolved by François and Hakim [45]. They compare the principles found by artificial evolution with biological networks and find some analogies.

Developmental models controlled by gene regulatory network imitate features of biological systems, e.g. scalability, self-organization, self-repair and robustness. Artificial GRNs are closer to biology than most other encoding methods and provide therefore the opportunity to compare biological results and behavior with artificial optimized results. Developmental plans and optimization strategies can therefore be adopted from biology, although these approaches are often more complicated than e.g. predefined developmental rules or direct encodings.

3.3 Evolution and Simulation of Animats

The simulation of artificial animals (animats) focuses on the morphology and the control of complete systems at a very basic level. By simulating and understanding a simple animal-like system, the longterm goal of building up to a human can be gradually realized [140]. Since the seminal work of Karl Sims, brain-body co-evolution has attracted much attention in the research field of artificial life [120, 131]. The most attractive aspect of the work is that a developmental model using a directed graph is adopted for both neural controller and body plan.

However, limited significant progress towards understanding biological principles have been made since Sims' work due to the following two facts. First, the power of the models for brain-body co-evolution remains practically unchanged [60, 98, 121]. A biologically plausible model should be able to describe the biological development of both nervous system and body plan. While models for either detailed modeling of neural development [76] or morphology (see Chapter 3.2) have been suggested, few models can achieve a balanced depth in modeling the development of both neural controller and body plan, and most of them are not able to perform biologically meaningful behaviors. Second, most work on brain-body co-evolution was meant mainly for improving the efficiency of generating a specific behavior, rather than understanding biological principles. An exception has been the work by Bongard and Paul [22], who studied the correlation between morphological symmetry and locomotive efficiency using a direct encoding.

Most recently, increasing efforts have been made to relate the research in brain-body co-evolution to biological principles. Jones analyzed the effects of the body plan on neural organization using energy constraints [70]. He found that a bilaterally symmetric body plan and neural architecture are favored during brain-body co-evolution of an elongated organism [68].

By taking energy efficiency into account in a hydra-like animat, he also showed that a ring-like neural structure emerged in the animat [69], which is similar to the nerve ring in biological hydra. The advantage of being able to evolve a bilaterally symmetric body plan or neural controller has been reported independently in [93, 105].

Although most models either use a complex model for the morphology and a simple one for the control or the other way around. But in those models, the genomes for both parts are separated. An exception is the work of Bongard who uses a gene regulatory model to develop locomoting animats or animats that should grow to touch an object using GRNs [20].

While most approaches only simulate their individuals or have a fixed morphology, there are two exceptions. The group of Lipson evolves the shapes and the control of robots in a simulated environment and they can automatically build the best individuals in real hardware to test their real performance [90]. They also evolve 'soft' robots with continuous and amorphous morphologies which can contract and expand themselves in order to move [56].

Meng et al. use self-reconfigurable modular robots controlled by a simple GRN so that the robot can adapt its shape to environmental change [96]. They also plan to build the modules in real hardware.

4 Model for the Evolution and Development of Animats

In the previous two chapters the biological and algorithmical fundamentals of this thesis have been described. This chapter contains a description of the model used. I start with the evolution strategy (ES), followed by the developmental GRN model. The chapter is concluded with the simulated mechanical interactions between different cells.

4.1 Evolution Strategy

In a standard ES, Gaussian mutations are applied to the real-valued parameters in the chromosome. Let k_i be a real-valued parameter encoded in the chromosome, a mutation operation changes its value as follows:

$$k_i' = k_i + z_i \tag{4.1}$$

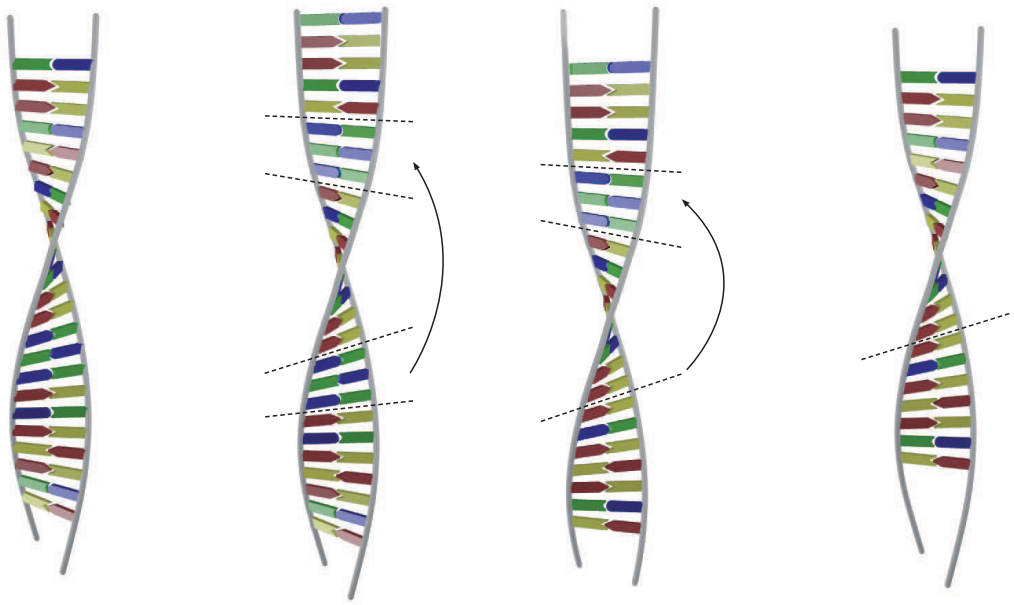
where

$$z_i \sim \mathcal{N}(0, \sigma_i). \tag{4.2}$$

In the above equation, σ_i is the strategy parameter. Either one σ for all values or an individual σ_i for each value k_i can be used. σ can be mutated or fixed.

In contrast to a standard ES, the algorithm used in this thesis has genetic variations such as gene duplication, gene transposition and gene deletion in addition to mutations. Gene duplication randomly copies a sequence of RUs and SUs in the chromosome and then inserts it, again randomly, into the chromosome. In the case of gene transposition or deletion, a random sequence is picked out of the RUs and SUs and is moved to another randomly chosen site on the chromosome, or is simply removed. Figure 4.1 shows a schematic diagram of genetic transposition and duplication.

Mutation is always performed, while gene duplication, gene transposition, and gene deletion is executed with a probability of p_{dup}, p_{trans} and



(a) Original DNA **(b)** Duplication **(c)** Transposition **(d)** Deletion

Figure 4.1: An example DNA is presented in a). Gene duplication, transposition and deletion of this DNA are shown in b), c) and d) respectively.

p_{del}, respectively. Gene duplication, transposition and deletion are exclusive, i.e. only one of them will be accomplished on the same chromosome in one generation. No recombination is used. The parameters k_i in the chromosomes are fixed to an interval of $k_i \in [0,1]$. In the experiments σ is fixed or limited to an interval, as given in the description of the experiments. If σ is limited to an interval, self-adaptation of σ is used:

$$\sigma_i' = \sigma_i \cdot e^{z_0 + z_{1,i}}, \tag{4.3}$$

where

$$z_0 \sim \mathcal{N}(0, \tau_0), \tag{4.4}$$
$$z_{1,i} \sim \mathcal{N}(0, \tau_1). \tag{4.5}$$

z_0 and $z_{1,i}$ represent a global and a local adaptation for each σ_i. τ_0 and τ_1 are small fixed values ($\sim 10^{-3}$).

The evolutionary algorithms employed in this thesis are part of the `Shark Machine Learning Library` [61, 118].

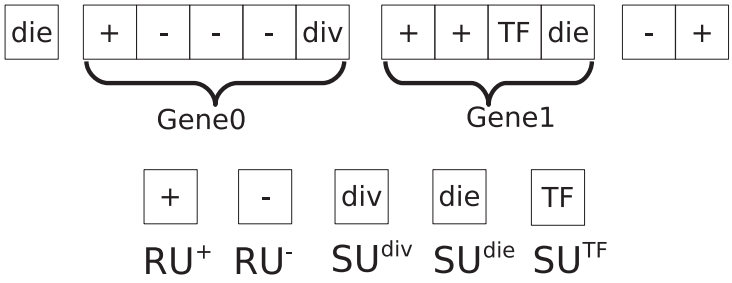


Figure 4.2: An example chromosome for the development.

4.2 Artificial Gene Regulatory Network

The morphological development simulated in this work is under the control of a gene regulatory network and physical cellular interactions. The morphological development starts with a single cell placed in the center of a two-dimensional computational area of size 100×80.

The GRN is defined by a set of genes, each consisting of a number of regulatory units (RUs) and structural units (SUs). SUs define cellular behaviors, such as cell division, cell death or the production of transcription factors (TFs) for intra- and inter-cellular interactions. Whether the SUs of a gene are expressed is determined by the activity level of the RUs of this gene. Note that single or multiple RUs may regulate the expression of single or multiple SUs and that RUs can be activating (RU^{+}) or repressive (RU^{-}), refer to Figure 4.2. The activation level of RUs is influenced by the TFs that can 'bind' to the RU. If the difference between the affinity value of a TF ($\mathrm{aff}_i^{\mathrm{TF}}$) and a RU ($\mathrm{aff}_i^{\mathrm{RU}}$) is smaller than a predefined threshold ϵ (in this work ϵ is set to 0.2), the TF can bind to the RU to regulate the gene activation. The affinity similarity ($\gamma_{i,j}$) between the i-th TF and j-th RU is defined by:

$$\gamma_{i,j} = \max\left(\epsilon - \left|\mathrm{aff}_i^{\mathrm{TF}} - \mathrm{aff}_j^{\mathrm{RU}}\right|, 0\right). \tag{4.6}$$

If $\gamma_{i,j}$ is greater than zero, then the concentration c_i of the i-th TF is checked whether it is above a threshold ϑ_j defined in the j-th RU:

$$b_{i,j} = \begin{cases} \max\left(c_i - \vartheta_j, 0\right) & \text{if } \gamma_{i,j} > 0 \\ 0 & \text{otherwise} \end{cases}. \tag{4.7}$$

Thus, the activation level contributed by the j-th RU (denoted by $a_j, j = 1,...,N$) can be calculated as

$$a_j = \sum_{i=1}^{M} b_{i,j}, \tag{4.8}$$

where M is the number of TFs that bind to the j-th RU. Assume the k-th gene is regulated by N RUs, the expression level (α_k) of the gene can be defined by

$$\boldsymbol{\alpha} = \boldsymbol{g}(\boldsymbol{c}), \tag{4.9}$$

$$g_k(\boldsymbol{c}) = 100 \sum_{j=1}^{N} l_j a_j (2s_j - 1),\ s_j \in [0,1]. \tag{4.10}$$

$\mathrm{sgn}(2s_j - 1)$ denotes the sign (positive for activating and negative for repressive) of the j-th RU and l_j is a parameter representing the strength of the j-th RU. If $\alpha_k > 0$, then the k-th gene is activated ($\delta_k = 1$) and its corresponding behaviors coded in the SUs are performed.

An SU that produces a TF ($\mathrm{SU}^{\mathrm{TF}}$) also encodes all parameters related to the TF, such as the affinity value, the decay rate D_i^c, the diffusion rate D_i^f. Which TF_i is produced is defined by the affinity value. The amount A_i of the TF_i is computed as follows

$$\begin{aligned} \boldsymbol{A} &= \boldsymbol{h}(\boldsymbol{\alpha}), \\ h_i(\alpha_k) &= \begin{cases} \beta \left(\frac{2}{1+e^{-20 \cdot f \cdot \alpha_k}} - 1 \right) & \text{if } \alpha_k > 0 \\ 0 & \text{otherwise} \end{cases}, \end{aligned} \tag{4.11}$$

where f and β are both encoded in the $\mathrm{SU}^{\mathrm{TF}}$.

A TF produced by an SU can be partly internal and partly external. To determine how much of a produced TF is external, a percentage ($p^{\mathrm{ext}} \in [0,1]$) is also encoded in the corresponding gene. Thus,

$$\Delta c_i^{\mathrm{ext}} = p^{\mathrm{ext}} \cdot A_i \tag{4.12}$$

is the amount of external TF to be produced and

$$\Delta c_i^{\mathrm{int}} = (1 - p^{\mathrm{ext}}) \cdot A_i \tag{4.13}$$

is that of the internal TF.

Table 4.1: Five real-valued parameters encoded in the regulatory units (RUs).

g_1	affinity value for the RU
g_2	the threshold for the concentration of the TF that reacts on the RU
g_3	sign: $[0, 0.5] \rightarrow$ inhibitory $(-)$ $[0.5, 1] \rightarrow$ activate $(+)$
g_4	strength l_j of the RU
g_5	-

External TFs are put on four grid points around the center of the cell, which first undergo a diffusion and then a decay process:

$$\text{Diffusion:} \quad \mathbf{u}_i^*(t) = \mathbf{u}_i(t-1) + 0.1 \cdot D_i^f \cdot (\mathbf{G} \cdot \mathbf{u}_i(t-1)), \tag{4.14}$$

$$\text{Decay:} \quad \mathbf{u}_i(t) = ((1 - 0.1 \cdot D_i^c)\, \mathbf{u}_i^*(t)), \tag{4.15}$$

where $\mathbf{u}_i$ is a vector of the concentrations of the i-th TF at all grid points and the matrix $\mathbf{G}$ defines which grid points are adjoining. The internal TFs underlie a decay process only:

$$c_i^{\text{int}}(t) = (1 - 0.1 \cdot D_i^c)\, c_i^{\text{int}}(t-1). \tag{4.16}$$

All internal and external concentrations of TFs are limited to an interval of [0,1]. The overall concentrations $\boldsymbol{c}^m$ of the TFs sensed by cell m is the sum of the concentrations of the internal TFs in cell m and the external TFs at the position of cell m

$$\boldsymbol{c} = \boldsymbol{c}^{\text{int}} + \boldsymbol{c}^{\text{ext}}. \tag{4.17}$$

The meaning of the values encoded in the RUs and SUs are described in Tables 4.1 and 4.2. Note, that all equations described in this section are the equations of cell m. To avoid too many indices, the index m is omitted and the equations need to be computed for each cell of an individual.

It is possible to put one or several prediffused, external TFs without decay and diffusion in the computational area. They can be constant or have a gradient in any direction.

The SU for cell division (SU^{div}) encodes the angle of division, indicating where the daughter cell is placed. A cell with an activated SU for cell death (SU^{die}) dies at the developmental timestep it is activated. When both cell

Table 4.2: Nine real-valued parameters encoded in the structural units (SUs). Values of $s_1 \in [0.6, 0.8]$ are used in Chapter 9 and 11 for neuron formation and $s_1 \in [0.8, 1]$ is used in Chapter 9 for cell movement.

	TF	divide	die
s_1	[0, 0.2]	[0.2, 0.4]	[0.4, 0.6]
s_2	-	-	-
s_3	-	-	-
s_4	TF affinity	-	-
s_5	decay rate	-	-
s_6	diffusion parameter	-	-
s_7	expression rate	-	-
s_8	f (see Equation 4.11)	division angle	-
s_9	externality	-	-

death and cell division are active on the same developmental step, only cell death is performed.

The model also provides the possibility of structural units for neuron differentiation ($\text{SU}^{\text{neuron}}$) or cell movement ($\text{SU}^{\text{move}}$), the $\text{SU}^{\text{neuron}}$ has different usages in Chapter 9 and Chapter 11, and is described there. SU^{move} is used and described in Chapter 9 only. Because some SUs are not used in some parts of this thesis, which means the SUs perform no action, it is possible to have genes without action.

Figure 4.3 shows a block diagram of the main components of a GRN in one cell, describing the cell dynamics. The cell dynamics can become coupled through external transcription factors, which underlie a diffusion and decay process and are position dependent. The number of TFs involved in gene regulation of the cellular behaviors is defined by the genome and the parameters in the resulting GRN as well. The number of cells also changes during development, it starts with one single cell and two external TFs. The maximum number of cells is limited to 700 cells to reduce computational cost. From a control system point of view, the developmental system is composed of a changing number of nonlinear dynamical sub-systems with a changing number of system states, and the dynamics of the sub-systems are strongly coupled with each other.

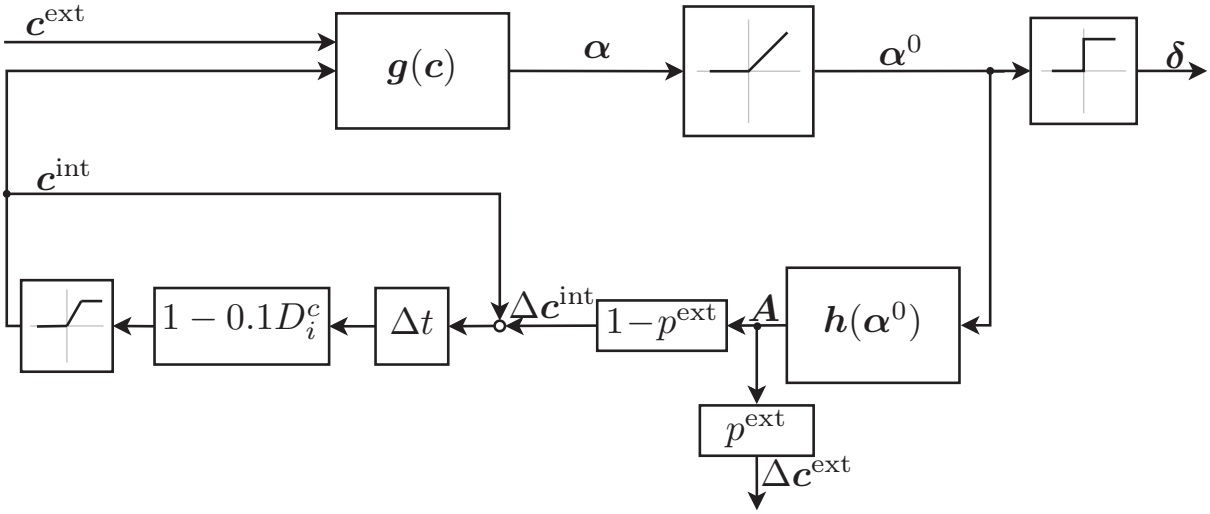


Figure 4.3: Block diagram of the model of a single cell.

4.3 Mechanics of the Cells

The morphological development starts with a single cell put in the center of a two-dimensional computational area. When the cell starts to divide, it is necessary to simulate mechanical interactions between the different cells, because their positions are not fixed to a grid. The nonlinear function f describes the forces between two cells (see Figure 4.4):

$$f(d) = \begin{cases} d - 2r + b & \text{if } d \leq 2r, \\ -e^{-3(d-2r)} + e^{-2.5(d-2r)} & \text{if } 2r < d \leq 5r, \\ 0 & \text{if } d > 5r, \end{cases} \tag{4.18}$$

where d is the distance between the centers of the two cells, r is the radius of the cells (here $r = 1$) and $b = 0.0022$. For distances that are smaller than the sum of the two radii, i.e. the cells overlap, the cells repel each other (negative forces). b is set to a small positive value (here $b = 0.0022$) so that cells that have a small overlap only adhere to each other. Cells that do not overlap, attract each other with decreasing forces with larger distances. For distances longer than five times the radii, there are no forces between the two cells.

When three or more cells are aligned perfectly on a straight line, the cells do not cluster despite the adhesion forces. To avoid this behavior and to make the model more plausible, a small random value is added to the positions of all cells, which results in a clustering of the cells that have been in one line.

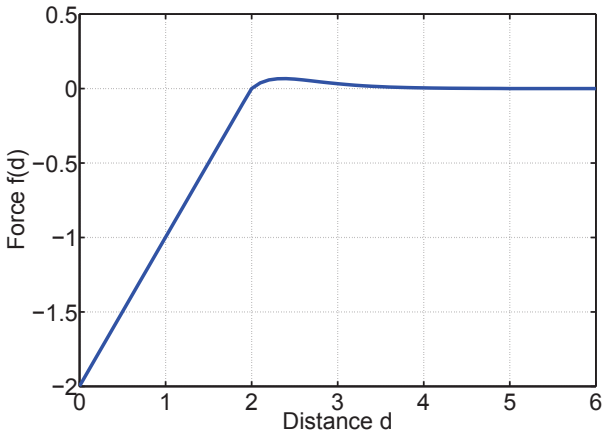


Figure 4.4: Force between two cells depending on the distance between their centers. The radius of each cell is 1.

The new positions of the cells are basically defined by a spring mass damper system:

$$\begin{aligned} F_x^i &= m\ddot{x}^i + c\dot{x}^i + kx^i \\ F_y^i &= m\ddot{y}^i + c\dot{y}^i + ky^i, \end{aligned} \tag{4.19}$$

where $\boldsymbol{F}^i$ are all forces on cell i. A detailed description of the equations and how they are solved is given in Appendix A.

Part I

Morphological Development

5 Evolution and Development of an Elongated Morphology

The experiments described in this chapter are the basis for the analyzes in the following three chapters. To imitate the morphology of e.g. *C. elegans* or *Hydra* in 2D, an elongated shape is evolved. In this chapter, the basic experiments of evolving an elongated morphology are outlined. Random individuals often result in too many or no cells, where the former is caused by a gene for cell division that is always active and the latter by an active gene for cell death at the beginning of the development. Individuals with an acceptable number of cells, which grow from a single cell result in most cases in a circular shape because of the adhesion forces and the random jittering of the cells (see Chapter 4.3). The following three chapters contain different analyzes on the development and the evolution based on the experiments presented in this chapter.

5.1 Basic Setup

The basic setup of the model and the evolution strategy is shown in Table 5.1.

The performance of the ES is similar with a fixed σ or with stepsize adaptation, so σ is fixed for these experiments. This was shown in experiments that are not described here. The individuals should have an approximated width-to-height ratio of $2a : 2b$, here $a_{max} = 5$, $b_{min} = 30$ and $b_{max} = 40$. This means the individual should be longer than the blue, dashed lines in Figure 5.1 but it should not grow larger than the black solid box. Thus, the fitness function is defined as follows:

$$\begin{aligned} f = p_1 - p_2 - \min\left\{\min_i\left\{y^i\right\}, -a_{max}\right\} \\ + \max\left\{\max_i\left\{y^i\right\}, a_{max}\right\}, \end{aligned} \tag{5.1}$$

Table 5.1: Basic setup of the used model

size of computation area	100×80
SUs	SU^{div}, SU^{die}, SU^{TF}
prediffused TFs	2 with gradients in x and y direction
max. developmental steps	15
μ	30
λ	200
elitists	3
σ	10^{-4}
p_{dup}, p_{trans}, p_{del}	0.05, 0.02, 0.03
max. generations	1000

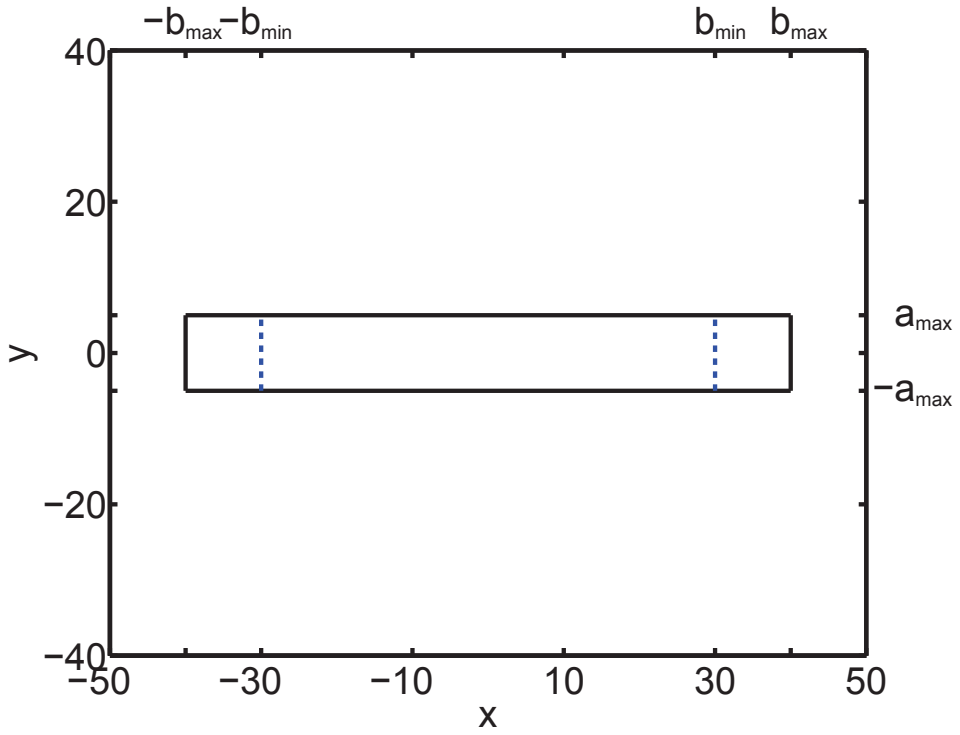


Figure 5.1: Optimal shape of the individuals. All cells should be connected and the individual should be longer than the blue, dashed lines. Cells outside the black, solid box are penalized.

where (x^i,y^i) represents the position of the i-th cell and

$$p_1 = \begin{cases} p_3 - \min_i \{x^i\} & \text{if } \min_i \{x^i\} < -b_{max} \\ -p_4 & \text{if} -b_{max} < \min_i \{x^i\} < -b_{min} \\ \min_i \{x^i\} & \text{otherwise} \end{cases}, \tag{5.2}$$

$$p_2 = \begin{cases} -p_3 - \max_i \{x^i\} & \text{if } \max_i \{x^i\} > b_{max} \\ p_4 & \text{if } b_{max} > \max_i \{x^i\} > b_{min} \\ \max_i \{x^i\} & \text{otherwise} \end{cases}, \tag{5.3}$$

$p_3 = 70$ and $p_4 = 30$. p_1 and p_2 depend on the length of the individual and the third and fourth term of Equation 5.1 rely on its width. To achieve a sensible yet computationally tractable size of body morphology, the number of cells (n_c) is constrained between 10 and 500. A penalty of $600 - n_c$ will be applied if $n_c < 10$ and a penalty of n_c if $n_c > 500$. If the cells in the developed morphology are not fully connected, a poor fitness of 50 will be assigned. Remember, all evolutionary optmizations in this thesis are minimization tasks and therefore a good individual is represented by low fitness values.

Two prediffused, external TFs without decay and diffusion are put in the computation area. The first TF has a constant gradient in the x-direction and the second in y-direction (see Figure 5.2). Although the aspect of preserving the predefined gradients would be very interesting to analyze in artificial evolution experiments (Holstein et al. [59] argue that molecular patterning systems could be directly subject to selection), the evolution of both together, i.e. the patterning system and the genetic response seems to be beyond the capabilities of current systems. Therefore, in these simulations, prediffused maternal gradients that guide the embryogenesis of the organism and their regeneration capacity are used (and artificially maintained). Maternal gradients in the early development of organisms have been observed consistently. In *Hydra*, head, foot and tentacle activators have been described by Meinhard [95].

5.2 Results

The resulting fitness curves are in Figure 5.3. Six out of 15 runs converge to the global optimum, in addition five of them find a solution close to the optimum, while four runs fail to find a good solution. The fitness curves

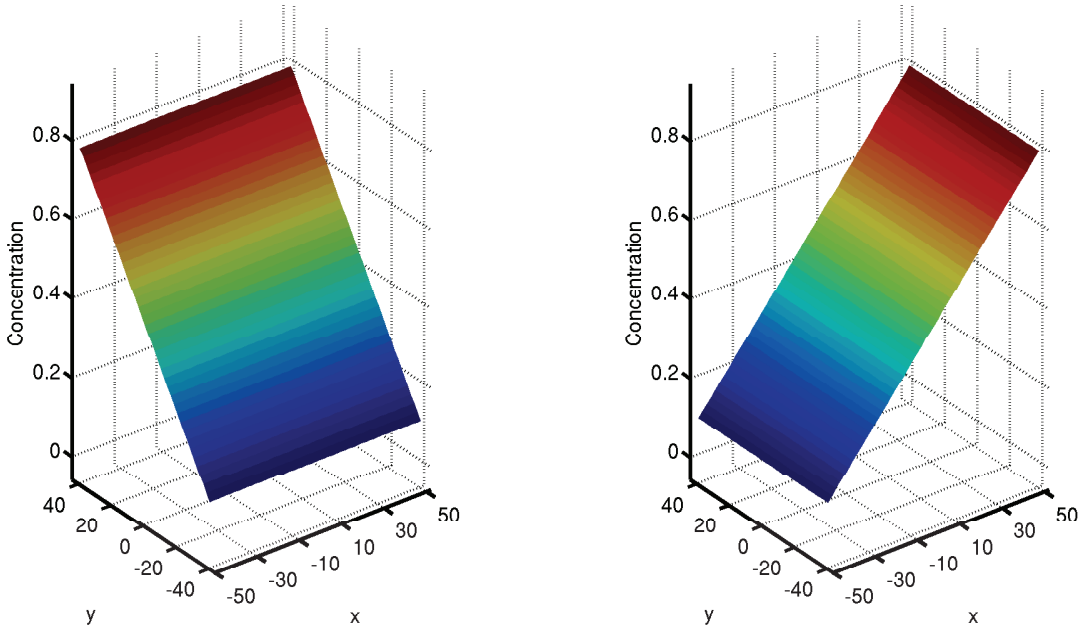


Figure 5.2: Concentrations of the prediffused TFs.

generally have many plateaus with (often huge) jumps in the fitness in between.

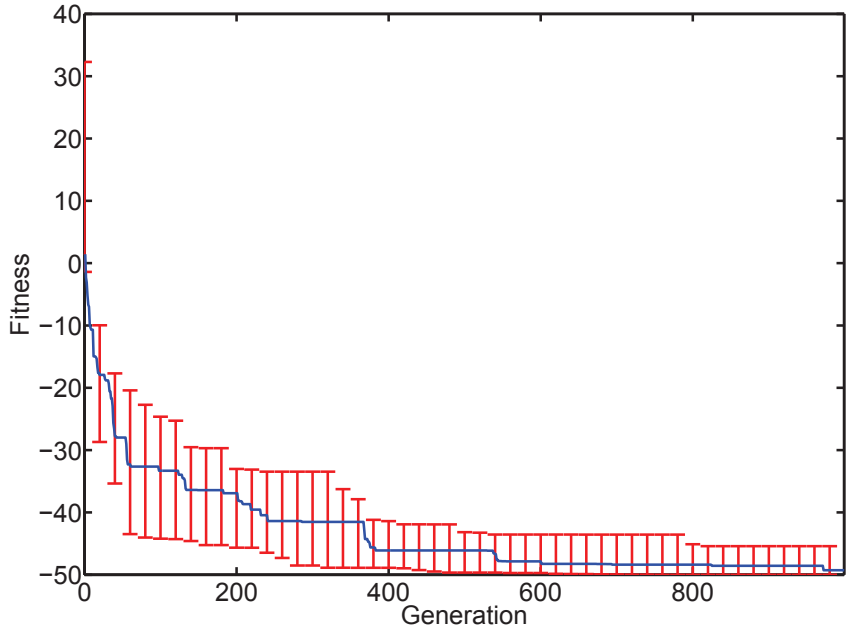
The development of one good individual is shown in Figure 5.4, the associated GRN in Figure 5.5. In Figure 5.6 only the connections of the GRN that are really used are shown.

For the development, we see a position dependent activation of the different types of genes. In the GRN we can see that only few of the existing connections are used during development.

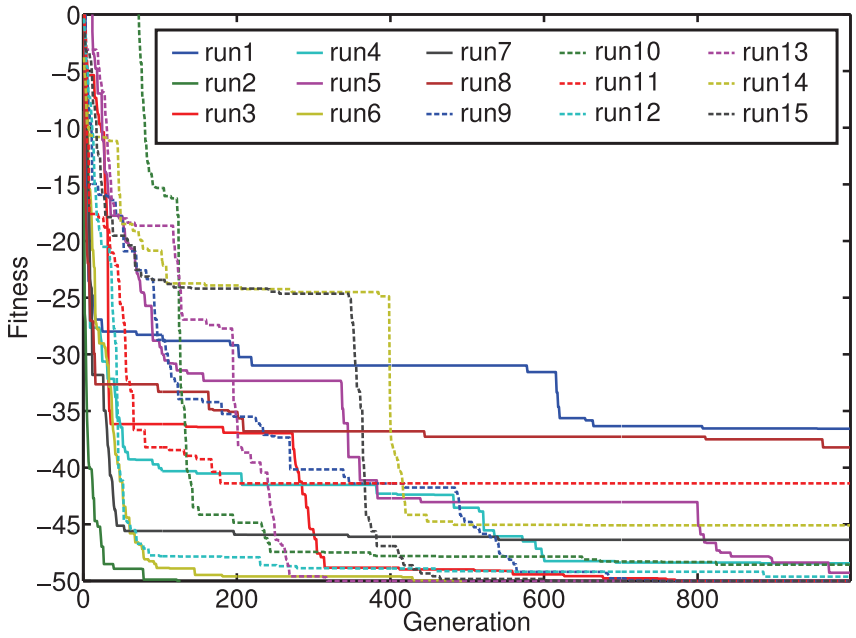
5.3 Discussion

Evolving an elongated morphology where the development is controlled by GRNs, is a non-trivial task, but the evolutionary process finds a good solution in most cases. The next three chapters contain different analyzes to increase the understanding on the development itself as well as on the evolution.

In this chapter, the development of the individuals is stopped after a fixed number of developmental steps (15 in this case). If developed longer, most individuals grow larger. A fixed number of developmental steps is simple, but biologically not plausible, because a huge advantage of developmental systems is their ability of self-repair, which needs a never ending



(a) Median with 25th and 75th percentiles.



(b) All fitness curves.

Figure 5.3: Fitness curves for all 15 runs to evolve elongated morphologies.

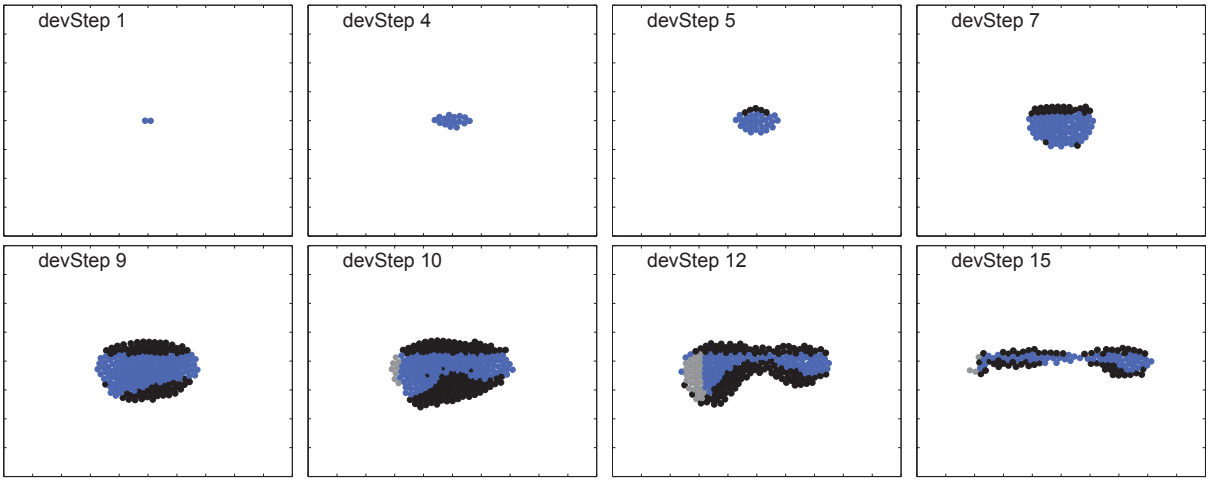


Figure 5.4: The development of the best individual of run 9. Cells that will divide in the next developmental step are marked in blue, dying cells in black and idle cells are marked in gray.

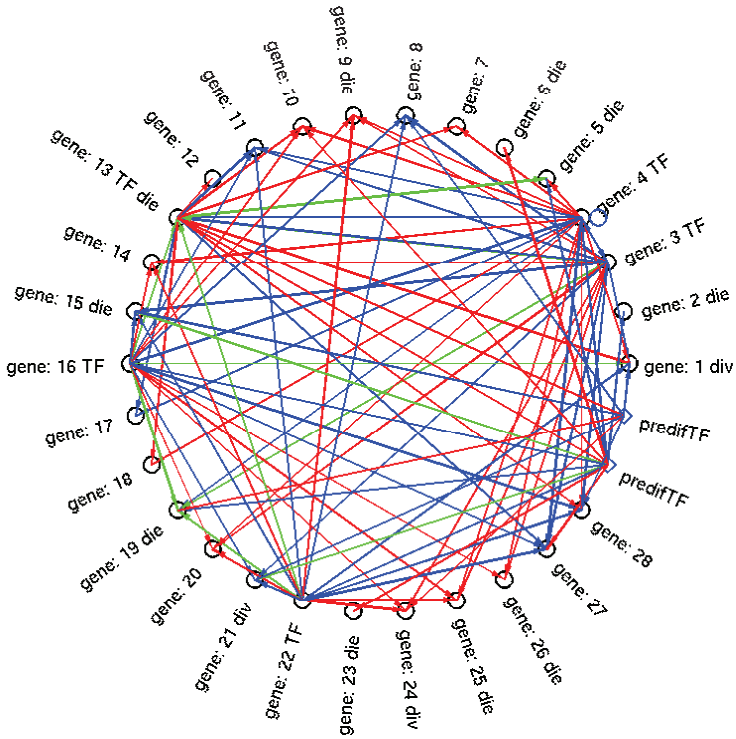


Figure 5.5: GRN of the best individual of setup 1, run 9. Dots represent the different genes, diamonds the two prediffused TFs. Arrows symbolize the connections between the different TFs and genes. All possible interactions coded in the genome are shown. Red arrows represent an inhibiting connection, blue an activating and green arrows represent both, an activating and inhibiting influence.

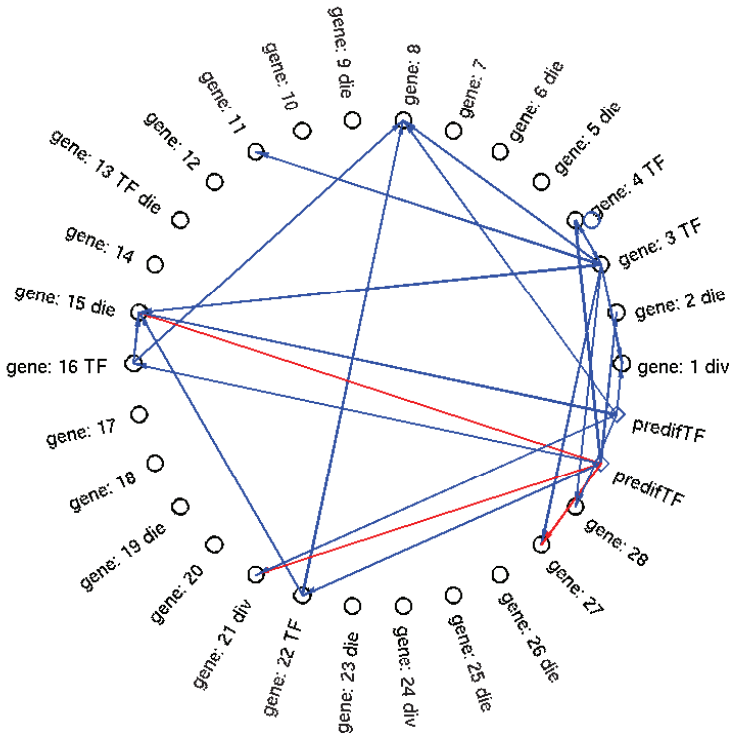


Figure 5.6: GRN of the best individual of setup 1, run 9. Only the interactions that are used during development are illustrated. For a detailed explanation see Figure 5.5.

and stable development. Then, the control of cellular functions by a GRN persists during the whole lifetime of an individual. I analyze the stability and self-healing abilities of the system in Chapter 6.

This is followed by two analysis of the evolutionary process itself, the first based on network motifs and their correlation to the jumps in the fitness curve (Chapter 7). As we can see in Figures 5.5 and 5.6, only few genes and connections between the genes are used during development. Therefore, in Chapter 8 an analysis on how the unused, redundant genes and connections influence evolutionary performance is provided.

6 Stability and Regeneration during Development

Developmental stability and regeneration are fundamental features of biological development. The cnidarian *Hydra* has the ability to regrow when cut into small pieces. The complex genetic interactions that lead to this stable growth and regeneration are difficult to analyze, especially from the evolutionary development perspective. The ability of stable growth and self-repair of the model presented in this thesis is analyzed in this chapter. Based on the experiments described in the previous chapter, dynamic stability has evolved successfully, in which a balance between cell division and cell apoptosis[1)]is reached. Some individuals can also regenerate themselves, so after removing some cells, the individual regrows. An analysis of the gene regulatory network is done and in most cases a close coupling between the predefined transcription factor and the activation of the genes for cell death, that is mainly responsible for the stable growth and shape formation, is found. This is consistent with the biological findings suggesting that morphological growth in early development of some organisms often depends on the maternal gradients. Furthermore, cell apoptosis will play an important role during the dynamically stable development of the artificial organism described in this chapter. In this simulation, the analysis is restricted to how the regulatory system responds to cell loss assuming that an appropriate signal is locally available.

Based on the model described in Chapter 4 and the experiments described in Chapter 5, the target of this chapter is to achieve dynamic stability and *Hydra*-inspired regeneration using physical cells that interact in continuous space. The outline of the experimental setup is given in Section 6.2 and the results and a discussion are given in Sections 6.3 and 6.4, where an analysis of the resulting GRN structure with respect to reoccurring patterns (motifs) during the evolutionary process is presented.

[1)] Cell apoptosis is the process of programmed death of a cell [31].

6.1 Stability and Regeneration in Developmental Models

Fleischer and Barr have been one of the first to discuss the issue of limiting growth in the context of artificial development using hand-coded genomes [44]. They use a threshold of the concentration of a chemical to disable cell division or a limited concentration of a chemical that is exhausted by cell divisions and cannot regenerate to limit the growth. For both solutions the ability of regeneration is difficult.

Miller [100] discusses the issue of stability in the context of artificial development in more detail. Instead of manually stopping the developmental process after a predefined number of steps, the target is to reach a steady state where the artificial system (like the French flag) maintains its morphology and/or function. This stable state can be static or dynamic. In the static case, cells do not divide anymore, either because of an evolved mechanism in the regulatory system or due to additional cellular properties like contact inhibition. If the steady state is dynamically stable, cell proliferation and differentiation still occur, however, they are compensated by cell apoptosis. This leads to a state of constant cell turnover like in the cnidarian *Hydra*. Of course a combination of both mechanisms is conceivable, although supporting biological evidence remains to be revealed.

Streichert et al. analyze the ability of limited growth and compare two models that simulate a GRN, a random boolean network (RBN) implementation and an S-system[2)]. They find that the RBN model converges faster but is more fragile to stochastic events [130]. They also show the ability of the RBN model to perform self-repair. In their model, cells can divide only if there is enough space, so cell division can be inhibited by neighboring cells.

6.2 Setup for Stable Growth and Regeneration

The developmental model defined in Chapter 4 for the development of cells controlled by a GRN is used to analyze stable development. Therefore, the basic setup defined in Chapter 5 is varied and three experiments are

[2)] S-systems (synergistic and saturable systems) which have been suggested by Irvine and Savageau, are based on nonlinear differential equations and are developed for the analysis of organizationally complex systems in biology [63].

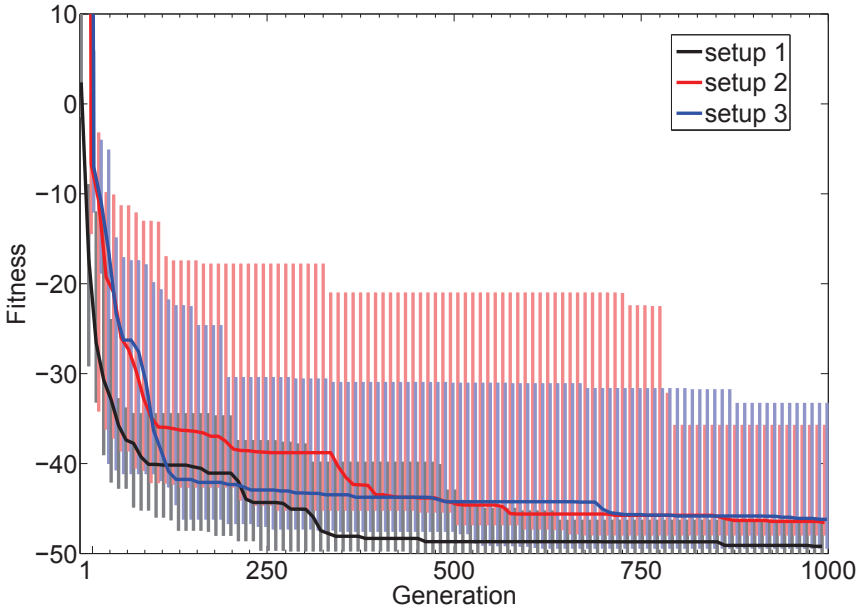


Figure 6.1: Boxplot (median and the 25th and 75th percentiles) of the fitness curves of the three setups. The fitness curves are given in Figure 6.2 - Figure 6.4.

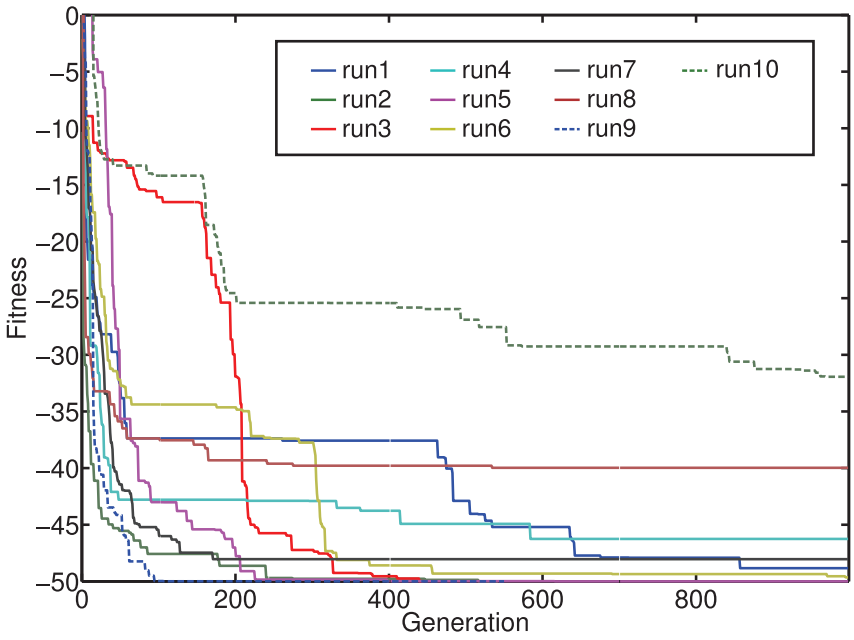


Figure 6.2: Fitness curves for ten runs with different random seeds for setup 1. Setup 1 is the basic setup with one fitness evaluation at the end.

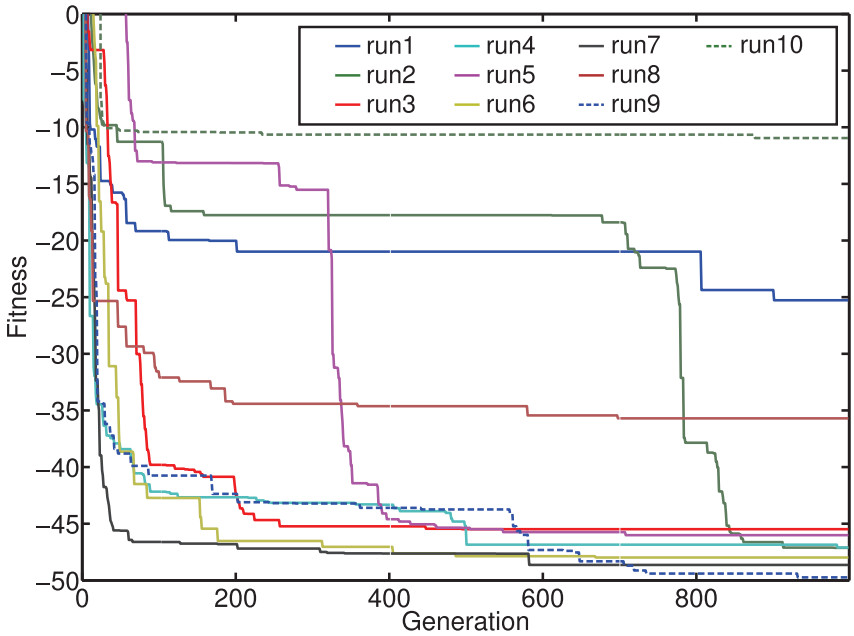


Figure 6.3: Fitness curves for ten runs with different random seeds for setup 2. The fitness in setup 2 is the mean fitness of ten different developmental steps.

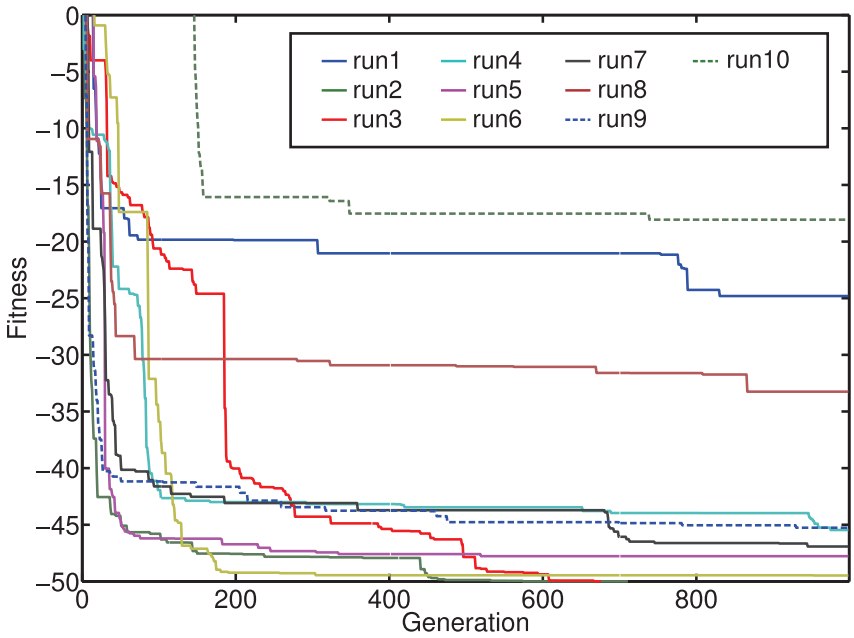


Figure 6.4: Fitness curves for ten runs with different random seeds for setup 3. The fitness is the mean fitness of ten different developmental steps and the individual is injured in-between.

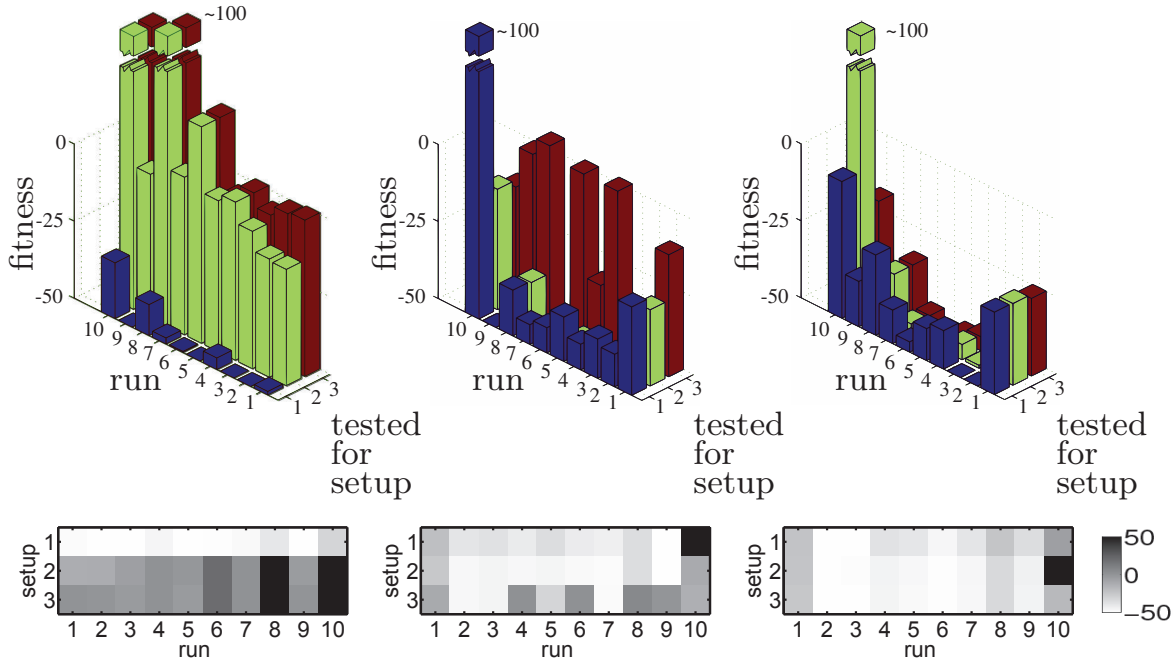
designed in order to test the system's ability for stable growth and regeneration. Equation 5.1 denotes the fitness at one developmental step. Three different evaluation procedures that compute the fitness at different developmental steps are defined. The first setup evaluates individuals once a certain number of developmental steps are completed. In the second setup, more developmental steps are simulated and the fitness is the mean of several evaluations after a certain number of developmental steps. In the third setup, the regeneration capacity of this artificial organism is tested by removing cells after a certain number of developmental steps:

- **Setup 1 (S1)**: Simulation of 16 developmental steps and evaluation after step 16. This is the same setup as in Chapter 5.
- **Setup 2 (S2)**: Simulation of 30 developmental steps and evaluations after steps 16,17,18,19,20 and 26,27,28,29,30. Thus, the fitness is the mean of the ten evaluations.
- **Setup 3 (S3)**: Same as setup 2, however, the individual is injured at its center once after step 20, so cells between $-5 < x < 5$ are removed.

According to the criteria in Equations 5.1 - 5.3 the best achievable fitness is -50; for setup 2 and 3 the same fitness is computed ten times and averaged. Ten trials with different random seeds for each setup represent the upper limit to what is computational feasible.

6.3 Evolutionary Results and Analysis

Figure 6.1 shows a boxplot (median and 25th and 75th percentiles) of all ten evolutionary runs for the three setups. The resulting fitness curves are shown in Figures 6.2 - 6.4. The mean fitness of S1 is better than those of S2 and S3, which confirms the assumption that the first setup is easier to evolve. The experiment using only 16 developmental steps (S1) shows, that four out of ten experiments reach the global optimum (-50), only one experiment fails completely with a fitness of -31.94. During the experiments without removing cells (S2) only one experiment reaches the global optimum, but still only 2 setups fail with a fitness worse than -40. The experiments with removing cells in developmental step 20 (S3) show that only two of the ten experiments reach the global minimum. Three runs result in a low performance.



(a) evolved for setup 1 **(b)** evolved for setup 2 **(c)** evolved for setup 3

Figure 6.5: The fitness of the ten best individuals of the ten evolutionary runs for each setup computed for all three setups. Note, that all values are truncated at +50.

To analyze the developmental stability and regeneration capability of the different evolved artificial organisms, the resulting individuals of all runs using all evaluation setups are compared. The results are shown in Figure 6.5. Figure 6.5a shows the fitness values of the ten individuals evolved using the fitness defined in S1 re-evaluated using the fitness function in S1, S2, and S3, respectively. Similarly, individuals evolved using the fitness function in S2 and S3 are re-evaluated using the fitness functions in the three setups and their fitness values are presented in Figure 6.5b and Figure 6.5c, respectively. Recall that the three setups are of increasing complexity and S2 includes aspects of S1, and S3 includes aspects of S1 and S2. An expectation is that individuals evolved in S3 can still perform well in S1 and S2 and individuals from S2 and S3 can perform well in S1. Indeed results in Figure 6.5 confirm this expectation, although the averaging over ten different developmental steps (in S2 and S3) slightly reduces the selection pressure to have a perfect individual after step 16, which is reflected by the slightly worse performance of individuals evolved for S2 and S3 on S1 as compared to individuals evolved for S1 on S1. As is evident from Figure 6.5a individuals evolved for S1 do not reach a dynamically stable state in

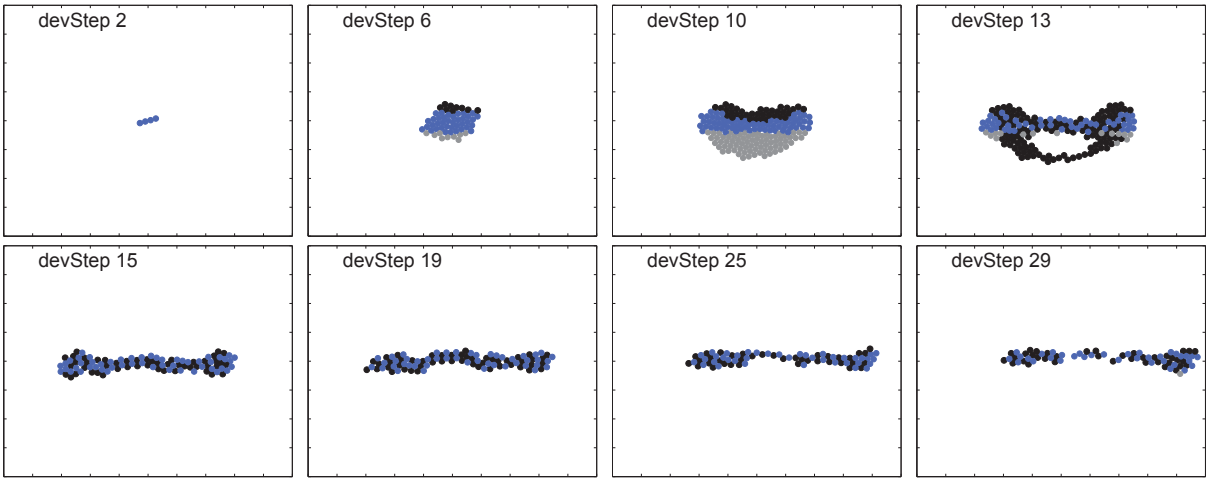


Figure 6.6: The development of the best individual of setup 1, run 9. Cells that will divide in the next developmental step are marked in blue, dying cells in black and idle cells are marked in gray.

any of the ten runs. Therefore, it seems to be more difficult to evolve dynamical stability than to tune individuals in such a way that they reach the perfect morphology exactly after 16 steps. From these experiments, a conclusion is that dynamically stable growth is evolvable only under an explicit selection pressure towards it.

At the same time, the observation from Figure 6.5b is that in three cases individuals evolved for S2 also perform well for S3. Thus, once the genetic mechanisms for reaching a dynamically stable state have been evolved, the regeneration capability follows almost without additional effort. This is in line with the observations from the regeneration capability of the cnidarian *Hydra*, which is closely connected to its enormous morphological plasticity. A dynamically stable state where cell proliferation and cell apoptosis are in a dynamic equilibrium represents a state of high morphological plasticity which in these experiments in three out of ten cases includes regeneration capability.

6.3.1 Analysis of the Development of Three Individuals

This section contains the analysis of the dynamics of the GRN and the structure behind the process of dynamically stable growth and regeneration for one individual that has been evolved for each of the three setups.

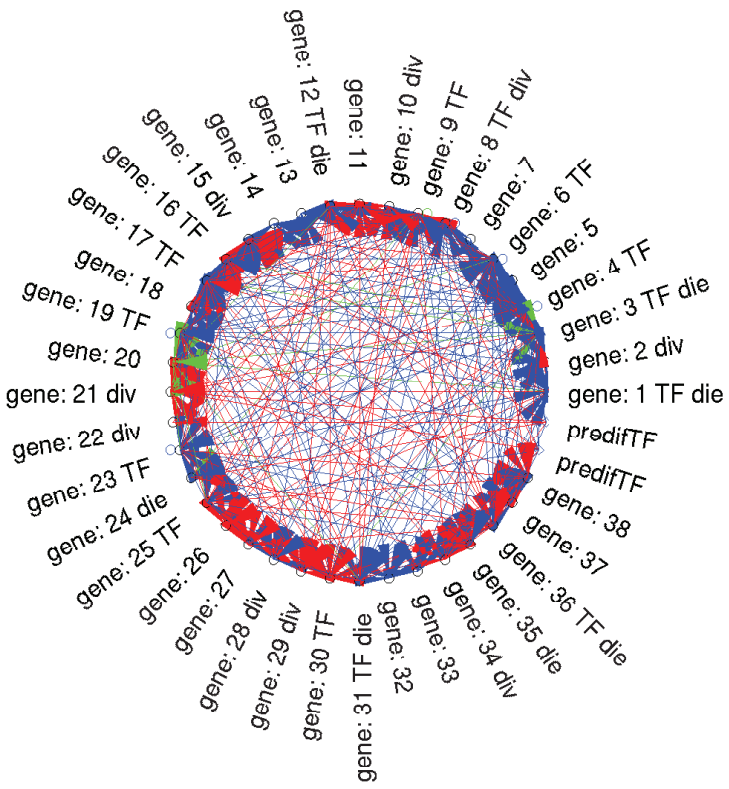


Figure 6.7: GRN of the best individual of setup 1, run 9. Dots represent the different genes, diamonds the two prediffused TFs. Arrows symbolize the connections between the different TFs and genes. All possible interactions coded in the genome are shown. Red arrows represent an inhibiting connection, blue an activating and green arrows represent both, an activating and inhibiting influence.

Of particular interest is what happens during the developmental process, e.g. which genes are used and how the genes are activated.

First Individual: setup 1, run 9

Figure 6.6 shows the cellular development belonging to the morphology of the best individual evolved from S1, run 9 (S1R9). At the beginning of the development, the cells in the middle (y-direction) divide and cells at the upper part die. The cells in the lower part perform no action. After developmental step 12 cell apoptosis becomes active in the lower part. At developmental step 15 cellular function changes and every second cell dies. (Note, that this individual is only evaluated at developmental step

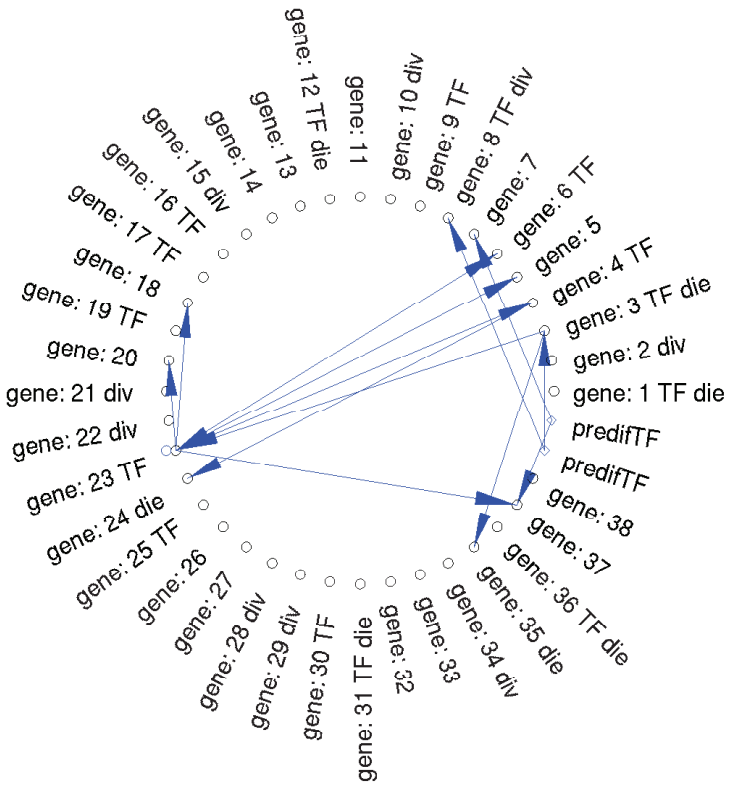


Figure 6.8: GRN of the best individual of setup 1, run 9, where only the interactions that are used during development are illustrated. For a detailed explanation see Figure 6.7.

16.) Then each cell first divides and then dies in the next developmental step.

The complete gene regulatory network shown in Figure 6.7 is highly connected. However, if all connections that are never activated during development are removed, only a few connections remain, as can be seen from Figure 6.8.

Remarkably, there are no inhibitory connections in the pruned GRN (the part of the GRN that is actually activated). We can see that only one gene is used for cell division (gene 8) and three genes for cell death (gene 3, 24, and 35). It is interesting to note the obvious high degree of redundancy in the GRN. This issue will be analyzed in Chapter 8.

Comparing the individual gene activations (data not shown) with the growing phenotype reveals that the gene for cell division is active in cells above a fixed x-position, see also Figure 6.6. As apoptosis is performed prior to cell division (if the GRN activates both functions in a cell) gene 3 (cell death) is also activated by the first predefined TF and active above

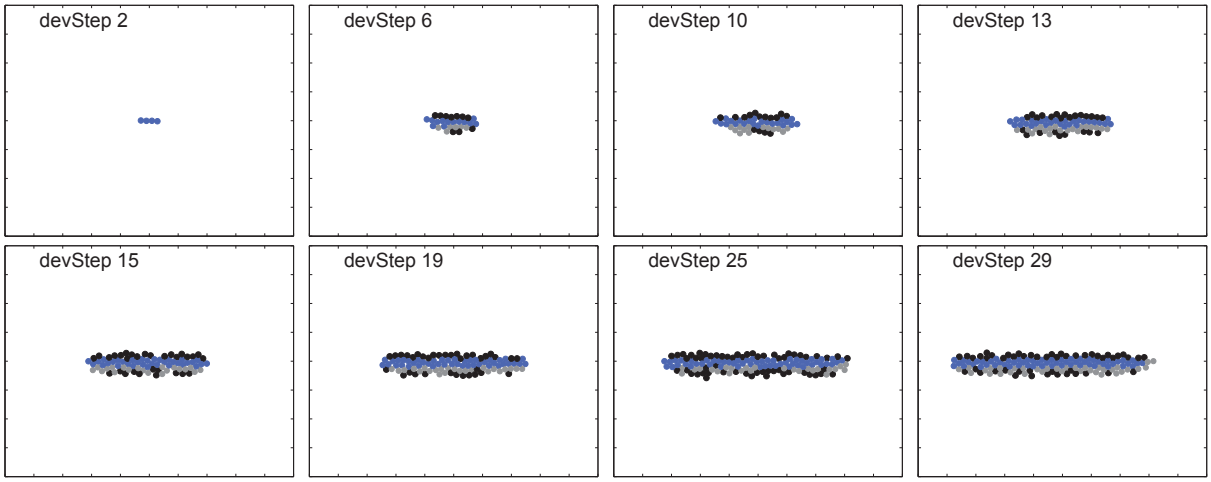


Figure 6.9: Development of the best individual of setup 2, run 2. For a detailed explanation see Figure 6.6.

a horizontal line. The activation of gene 35 is coupled to the activation of gene 3, it has no further influence on the development. The activation of gene 24 is more complex, it depends on the activation of gene 4 that itself depends on gene 23. The activation of gene 4 starts at developmental step ten, the concentration of the TF produced by gene 4 is high in the middle of all cells producing it. Every cell needs two developmental steps to produce enough of the TF to activate gene 24.

Second Individual: setup 2, run 2

The second example is the best individual from the second evolutionary run on S2, whose development is shown in Figure 6.9. Cellular function is position dependent. All cells above and below a defined y value die, cells in between divide, which is a simple but effective way of achieving a dynamically stable state in the y-direction.

After removing all unused connections the pruned GRN shown in Figure 6.10 can be analyzed. Only one gene (15) represents cellular division, which is activated by the first predefined TF and deactivated by the second. Therefore, gene 15 is active above a horizontal line, but deactivated at the right side of the individual. Genes 7 and 6 (cell apoptosis) are activated above and below a horizontal line, respectively.

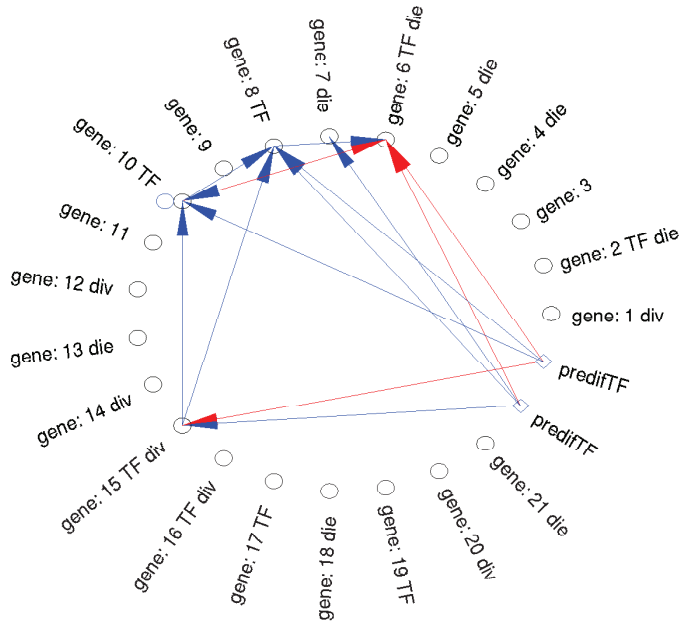


Figure 6.10: Pruned GRN of the best individual of setup 2, run 2. For a detailed explanation see Figure 6.7.

Third Individual: setup 3, run 2

The development of the third example individual from S3 is shown in Figure 6.11. The development is comparable to the second individual, but the pruned GRN is different, see Figure 6.12. Since the pruned GRN is still rather complex, it is further simplified by removing all connections that do not influence the fitness of the individual even though it is activated during development. The pruned and simplified GRN is shown in Figure 6.13. Gene 13 and genes 1, 9 and 20 represent cellular division and apoptosis. Gene 13 (division) is activated by the second predefined TF and is active in all cells except the far left ones. Gene 1 (apoptosis) is also activated by the second predefined TF and is only active in cells at the right hand side of the individual. Gene 9 (apoptosis) is active at the bottom of the individual. It is activated by the TF produced by gene 15 and deactivated by the first predefined TF. Gene 15 is active in all cells. Above a horizontal line, the TF of gene 20 activates it. Gene 20 (apoptosis) itself is also only active at the top and activated by the first predefined TF.

Next, some cells at developmental step 20 are deleted in order to observe the regeneration process. In Figures 6.14a, 6.14b, 6.14c, cells on the left, center and right of the individual are removed.

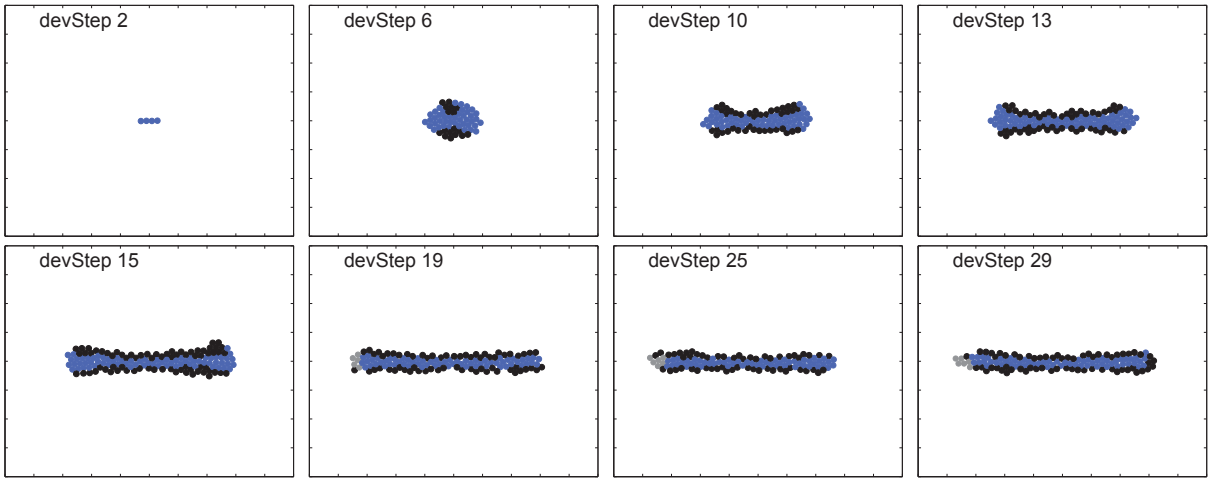


Figure 6.11: Development of the best individual of setup 3, run 2. For a detailed explanation see Figure 6.6.

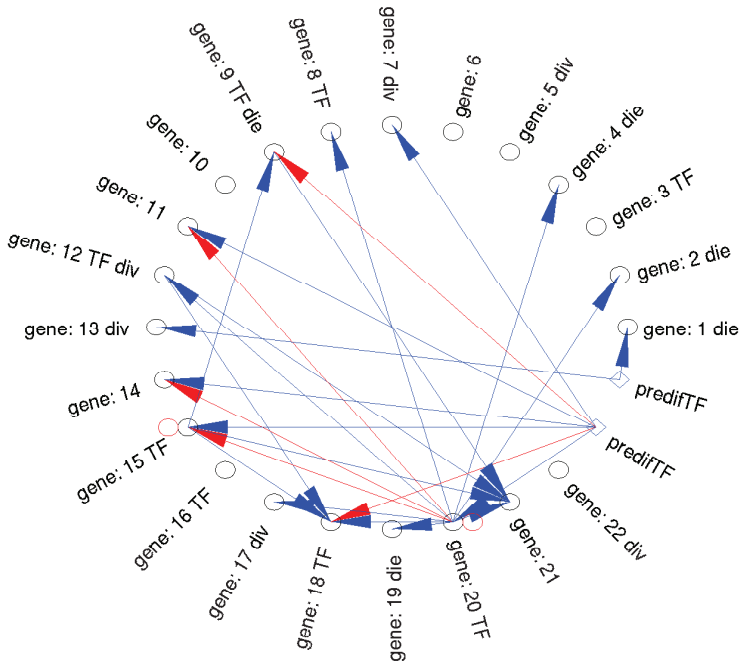


Figure 6.12: Pruned GRN of the best individual of setup 3, run 2. For a detailed explanation see Figure 6.7.

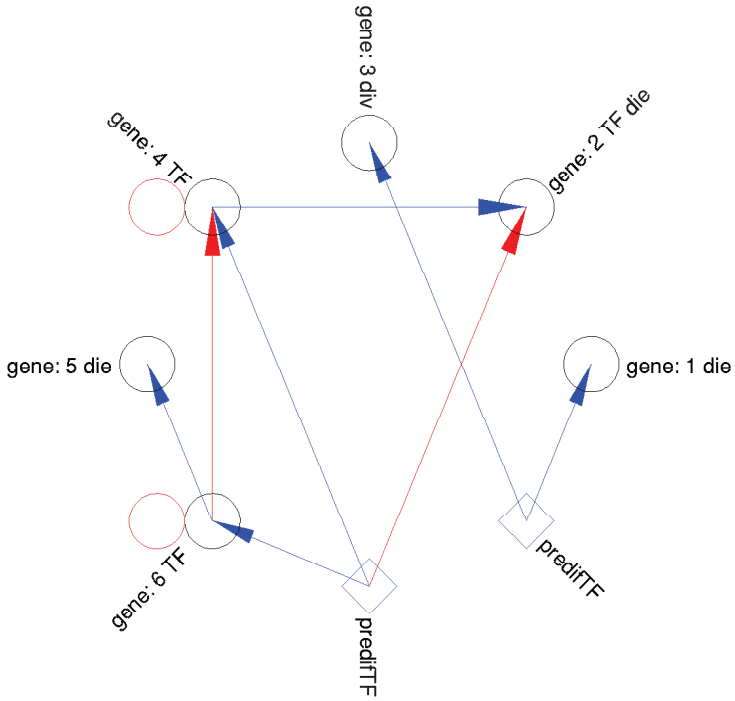
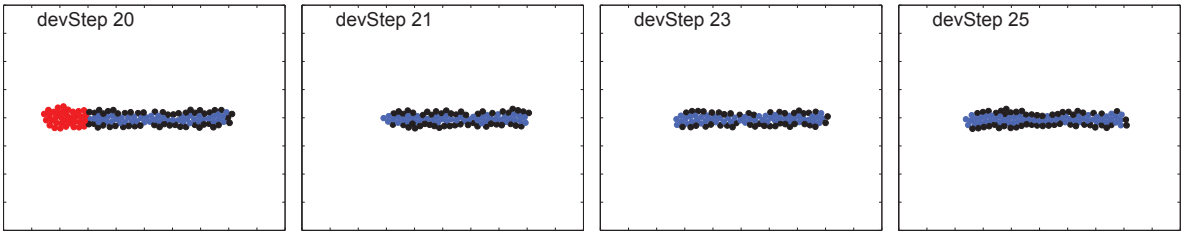


Figure 6.13: Pruned and further simplified GRN of the best individual of setup 3, run 2. For a detailed explanation see Figure 6.7.

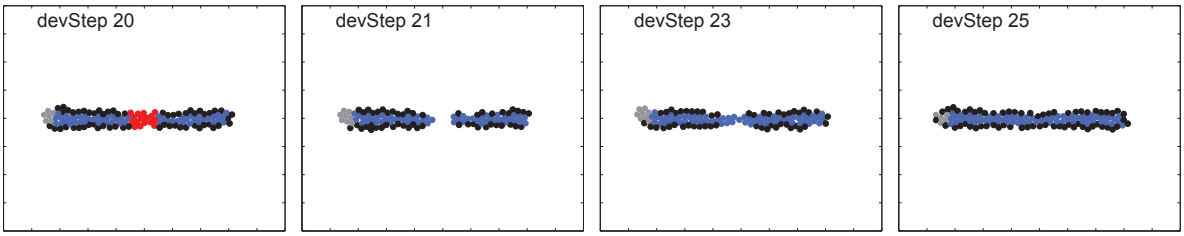
Figure 6.14 shows that in all three cases the regeneration process takes about five developmental steps. Although only deletion in the center of the artificial organism was included in the evaluation in S3, the resulting GRN function works well in all three cases. The high morphological plasticity is not locally confined and therefore, it does not matter where cells (or how many cells) are removed. Basically, the whole organism can be recovered from only a few cells – similar to the cnidarian *Hydra*.

6.3.2 The Role of Morphogen Gradients and the Evolution of Motifs

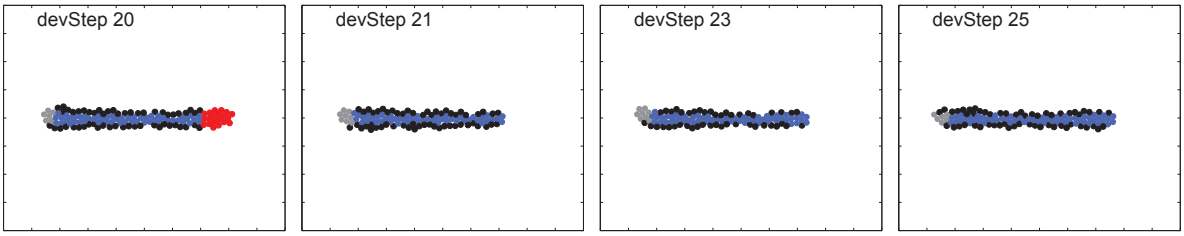
In most of the individuals, cellular growth is structured by controlling cell apoptosis through predefined (morphogen gradients) or produced transcription factors. Therefore, first the activations of an apoptosis-gene above the individual (where y is between 25 and 45) are counted, and which is termed motif one (M1). The activation can be directly regulated through the predefined TF or indirectly through a TF of a gene that is activated by the predefined TF. Of course deeper cascades are possible, however, mutational stability is reduced. The second motif (M2) performs



(a) deletion of cells with $x < -20$ at $t = 20$



(b) deletion of cells with $-5 < x < 5$ at $t = 20$



(c) deletion of cells with $x > 20$ at $t = 20$

Figure 6.14: Regeneration: Development of the best individual of setup 3, run 2. Some cells at the end of developmental step 20 at different positions are deleted: a) left, b) middle, c) right side of the individual.

the same function, i.e., controlled cell apoptosis on the right hand side of the individual (x between 0 and 10).

The plots in Figure 6.15 show the fitnesses of the different individuals and the occurrences of M1 and M2, where m_1 is the number of used paths activating an apoptosis gene at $25 < y < 45$, and m_2 is the number of used paths activating an apoptosis gene at $0 < x < 10$. The number of motifs are scaled in Figure 6.15, so $10m_1$ and $(10m_2 - 20)$ are plotted. M1 occurs in most runs early during evolution and its appearance is often accompanied by a fitness increase. If M1 is not present, either the fitness is low like in the individuals of setup 1, run 10 (S1R10), S2R10, S3R1 and S3R10, or an alternative mechanisms has evolved like in S2R3, S3R5 and S3R9:

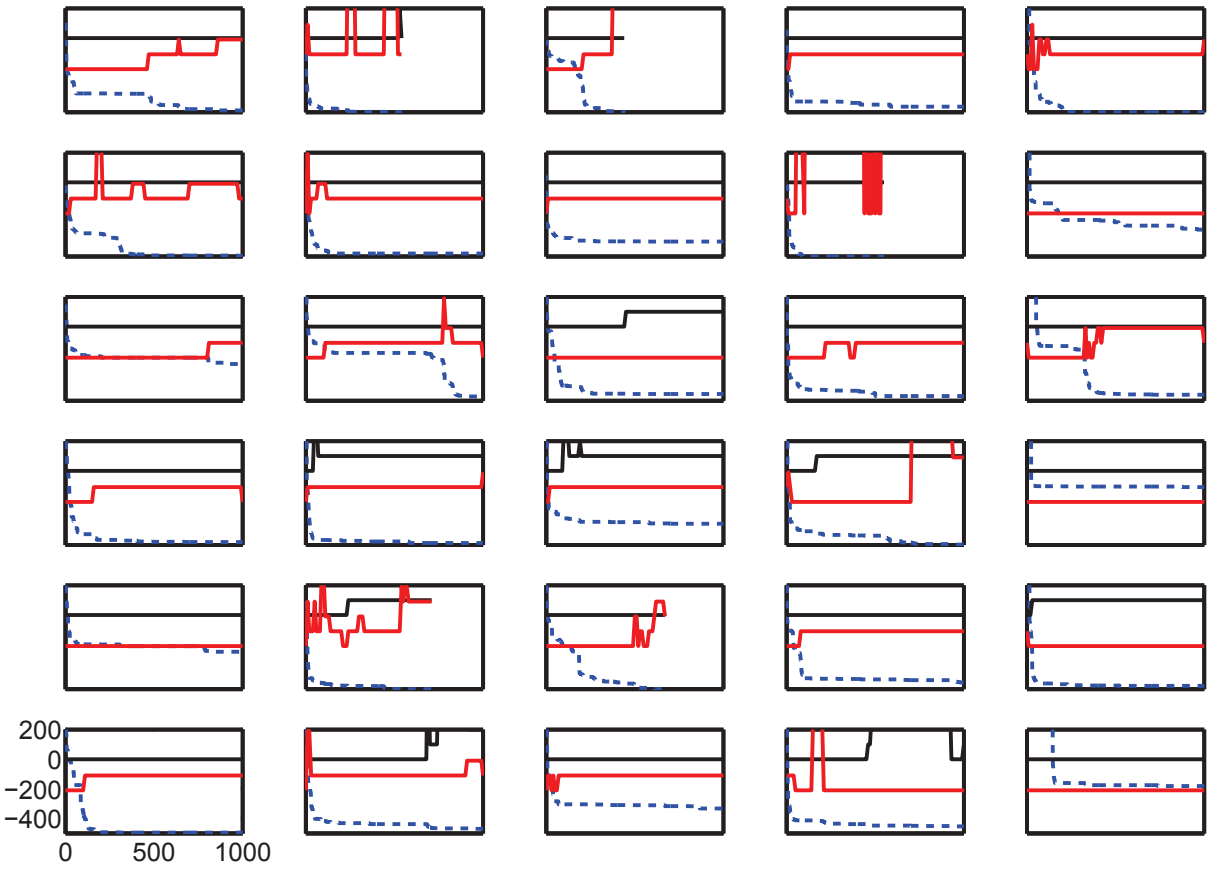


Figure 6.15: Activations of the morphogen gradients. First two rows: setup 1, third and fourth row: setup 2, last two rows, setup 3. For all figures: x-axis: generation, blue,dashed: Fitness, red, solid: apoptosis gene activated at the top of the individual (M1) (scaled to $10m_1$, see text), black, solid: apoptosis gene activated at the right side (M2) (scaled to $10m_2 - 20$, see text). The motifs are counted every tenth generation.

- S2R3: apoptosis is activated by an internal TF that is always produced. The cells that divide never reach the threshold of the TF to activate apoptosis because during division the concentration of the internal TF is halved.
- S3R5: cells divide only at the left and right ends of the individual, so no apoptosis is required in y-direction. The artificial organism does not reach a dynamically stable state because growth on the left side is uncontrolled. M2 controls cellular growth on the right side of the individual.
- S3R9: at the beginning of the development a blob forms, followed by an elongated shape. Most cells first divide and then die in the next developmental step, comparable to the first analyzed individual of the previous section.

The second motif does not occur in S1, because after 16 developmental steps the individuals have not reached the left/right border of the morphological fitness constraint. In S2 and S3, four out of ten runs evolved the second motif, whose occurrence is often accompanied by a fitness increase, e.g. in S1R3, S1R4, S2R5, S2R7, S3R4, S3R5, and S3R6.

Therefore, controlled cellular growth leading to a dynamic equilibrium of high morphological plasticity and regeneration capability is governed mostly by the interaction of predefined TFs (morphogen gradients) with cellular apoptosis and in fewer cases with cellular division. In all cases, cellular apoptosis is position dependent.

6.4 Discussion

Without having to rely on additional functions like contact inhibition or additional information like number of surrounding cells, it was possible to evolve individuals that reach a dynamically stable state of high morphological plasticity during development. In this state, cell turnover is in equilibrium, i.e., cell proliferation is of the same order as cell apoptosis. Dynamic stability could only be reached after distributing the evaluation of each individual over a number of developmental steps (ten in these experiments).

The regeneration ability of the artificial organisms often is a result (three out of ten cases) of the high morphological plasticity. One possible explanation is that once morphological plasticity has been evolved but regeneration is not yet realized, additional evolution of the functional regulatory

networks to enable regeneration is straightforward or at least easier than discovering dynamically stable growth in the first place.

The detailed analysis of the genetic regulatory networks revealed that most individuals achieve dynamically stable growth and regeneration through the position dependent control of cellular apoptosis using prediffused transcription factors (morphogen gradients). Unlimited growth and controlled cell removal leads to high morphological plasticity and to the desired regeneration capability. However, at the expense of high energy costs, which was not taken into account in these experiments. I identified two motifs that occur in the most successful artificial organisms that represent such a control.

In other models, as described in Chapter 3 contact inhibition or additional information like number of surrounding cells are used. In the model of Andersen et al. a GRN model based on the approach by Eggenberger is used and a stable development and a capacity of self-repair is achieved [5]. The cells are fixed on a grid, they use contact inhibition, so a cell surrounded by other cells is not able to divide. In contrast to Andersons work, my cells are not fixed on a grid, they can push each other, so they are always able to divide. They do not use cell death, opposing to the approach presented here. Basanta et al. evolve static and dynamic stability using a model based on **cellular automata** [12]. Some of their individuals perform self-repair. Cells can divide, move or die. Therefore, the basic ingredients like cell position or the number of cellular divisions is prespecified in order to enable convergence to a stable state.

The framework and the experiments presented in this chapter have been inspired by the amazing regeneration capability of the cnidarian *Hydra*. Since this model of an artificial organism is of a high abstraction level, the relation of these experimental results to biological observations has to be done very carefully. Nevertheless, from the biology of *Hydra* it is known that its regeneration capability is related to both its development and its high morphological plasticity. Furthermore, pre-patterning in *Hydra* is a prerequisite to successful regeneration. It is not yet known whether this signal is excitatory or inhibitory and how it is self-generated after lesion.

In this framework, the important question of signal generation by relying on the prediffused transcription factors to guide both the developmental as well as the regeneration process is circumvented. One next step would be to make the prediffused TFs time dependent and dissolve them after a number of developmental steps. This is likely to make stable growth based on controlled cellular division (as opposed to apoptosis) easier, because a time-dependent signal is available with information that can be used to

stop division. At the same time, the regeneration process will be more difficult because the GRN has to evolve a mechanism to self-organize pre-patterning after lesion to prepare for regeneration. First experimental results that have been conducted suggest that indeed the evolution of such a mechanism will be non-trivial.

It would also be interesting to ascertain whether evaluating the performance after e.g. step 16, 18 and 20 and taking only the best result would drive the evolutionary process toward dynamic stability. This way, one would find out whether it is difficult for evolution to discover dynamic stability or to tune it to exactly one developmental step.

This chapter contains an analysis of the development of some individuals, in the following chapter the network motifs and their occurrence during evolution are studied.

7 Evolved Network Motifs in GRNs

Among other research efforts, analyzes of small, frequently occurred network structures, often known as network motifs, have attracted much interest. However, the analysis of motifs on an evolutionary scale requires the data of many individuals from different evolutionary stages. These data are (currently) not available in biology. Therefore, it seems advisable to support the biological analysis with the results from computational models. Even though these models are usually abstract and the analysis is computationally expensive, it is the target to identify patterns that relate the emergence of motifs to the evolutionary progress in computational models.

Kashtan and Alon show the evolution of modularity and network motifs while switching between two fitness functions with common subgoals during evolution [73]. They also showed, that the overall evolution is faster than the evolution without switching the fitness.

Steiner et al. showed that the emergence of a negative feedback motif helps to enhance the mutational robustness [126, 128]. In this Chapter, an analysis of motifs in the GRNs in the best individuals of the whole evolutionary run are presented to see how various network motifs have contributed to the evolution of cellular development. The goal of the evolution is to obtain an elongated shape resulting from the cell growth process controlled by the GRN. The target shape is defined in Equations 5.1 - 5.3 and Figure 5.1. The changes in the number of motifs during evolution are analyzed and accordingly a run with some large jumps and a good solution at the end seems to be good for analysis. Therefore the evolutionary setup described in Chapter 5 with run 5 is examined. Additionally, the difference in the structure of the GRNs of two related individuals before and after a fitness jump are analyzed.

The next section contains an introduction to the widely studied network motifs. Then the experimental results of the evolutionary runs together with the number of motifs during the evolution are presented. The chapter

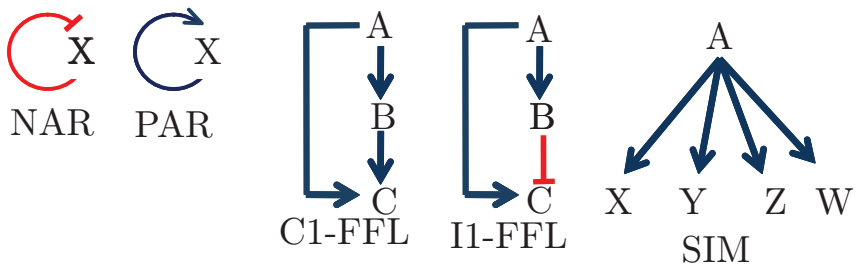


Figure 7.1: Network motifs (adapted from [3]). A, B, C, X, Y, Z, and W represent nodes of a network, e.g. they can stand for genes. Blue arrows indicate an activating connection, red arrows an inhibition.

is concluded with an analysis of two individuals followed by a discussion. Parts of this chapter are based on [115].

7.1 Static and Dynamic Network Motifs

Network motifs are sub-networks that occur more often in biological gene regulatory networks than expected at random (see Chapter 2.3). In this chapter, the occurrence of different types of regulatory motifs, such as autoregulation, feed-forward-loops and single input modules are analyzed, see Figure 7.1. In the following, the function of a few network motifs is described, see [3, 4]:

- **Negative autoregulation (NAR)** defines a gene whose product directly inhibits its own expression. Such motifs can speed up the response time compared to a gene without NAR with the same steady state. It leads to steady states with a rapid rise and a sudden saturation. NAR also promotes robustness.
- The **positive autoregulation (PAR)** slows down the response time and can lead to bi-stability.
- The **coherent feed-forward loop 1 (C1-FFL)** results in a fast convergence to a steady state but a slow decrease of the concentration.
- The **incoherent feed forward loop 1 (I1-FFL)** can act as a pulse generator. It can turn a concentration very fast on with an overshoot, and then it converges to its steady state.

- The **Single input module (SIM)** consists of one gene regulating many other genes. Temporally sequential cellular events can be controlled with a SIM.

There are a lot of different FFLs, among which C1-FFL and I1-FFL are the most frequent ones in E. coli and yeast. The functional analysis described above is performed on isolated motifs, and therefore their behavior in a whole network can be very different.

All possible connections of a GRN define the *static network*. Therefore, the *static network motifs* are all possible network motifs regardless of whether they are actually used during cell operations. In this chapter, only the network connections that are really used during development, which constitute the *dynamic network*, are analyzed. The related motifs are then termed the *dynamic network motifs*. In order for a static motif to be counted as a dynamic motif, all motif connections have to have been activated (above the threshold) in at least one cell at anytime during development. Thus the dynamic motif must play an active role during cell operations and not just a potential role as the static motif. Of course dynamic motifs are a subset of static motifs.

7.2 Results

The best and mean fitness curves of an evolutionary run are presented in Figure 7.2. There are two fitness jumps around generations 350 and 800 during the whole evolution. The resulting shape of the best individual in the last generation is shown in Figure 7.3. The morphologies of the individuals of the first generations all result in either no cell or too many cells (the runs with more than 700 cells were aborted). In Figure 7.4 the total number of genes is shown. The number of genes is nearly constant, except for one huge jump at the end of the evolution.

7.3 Analysis of Dynamic Network Motifs

The different network motifs for all selected individuals every 5th generation are counted. The motifs of the best individual and the mean of the parent generation are presented in Figures 7.5 - 7.8. The algorithm counts all occurrences of one gene activating two others as one SIM (which is then a three node motif). When there is one gene activating more than two

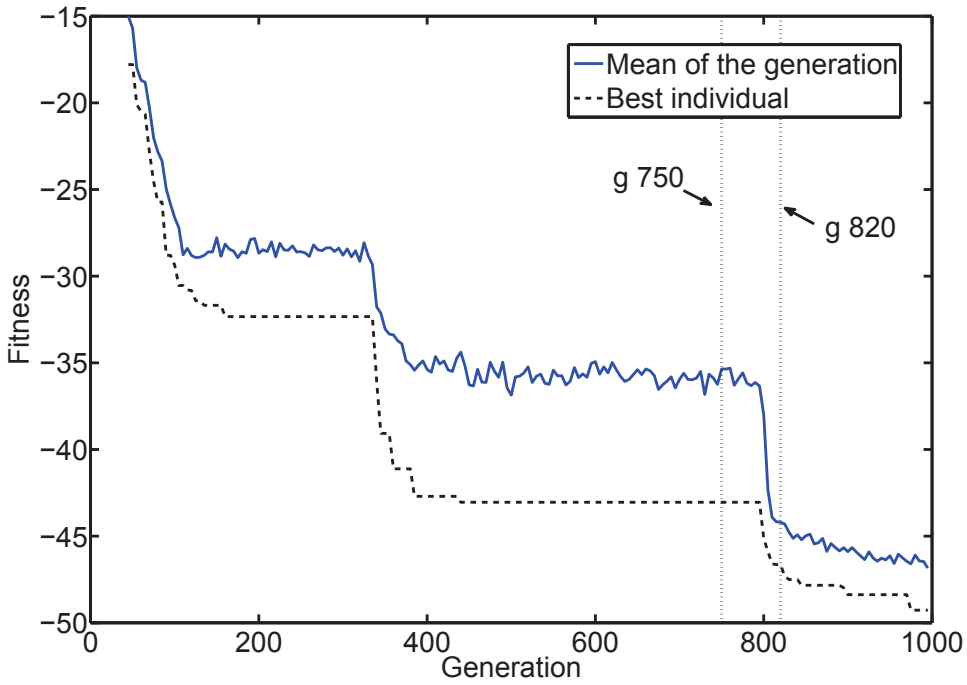


Figure 7.2: Fitness curves of the analyzed evolutionary run. Solid line: mean of the generation. Dotted line: best individual.

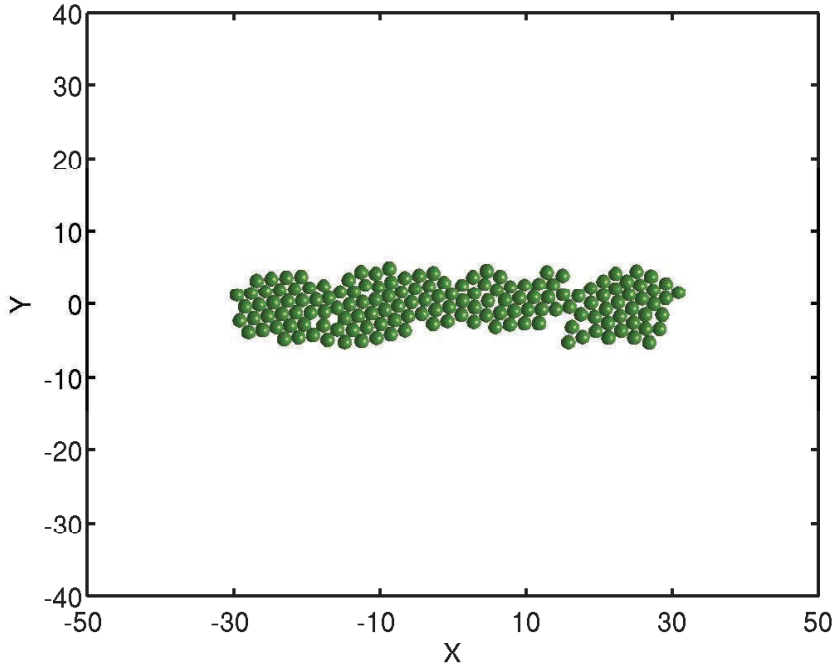


Figure 7.3: Resulting shape of the best individual.

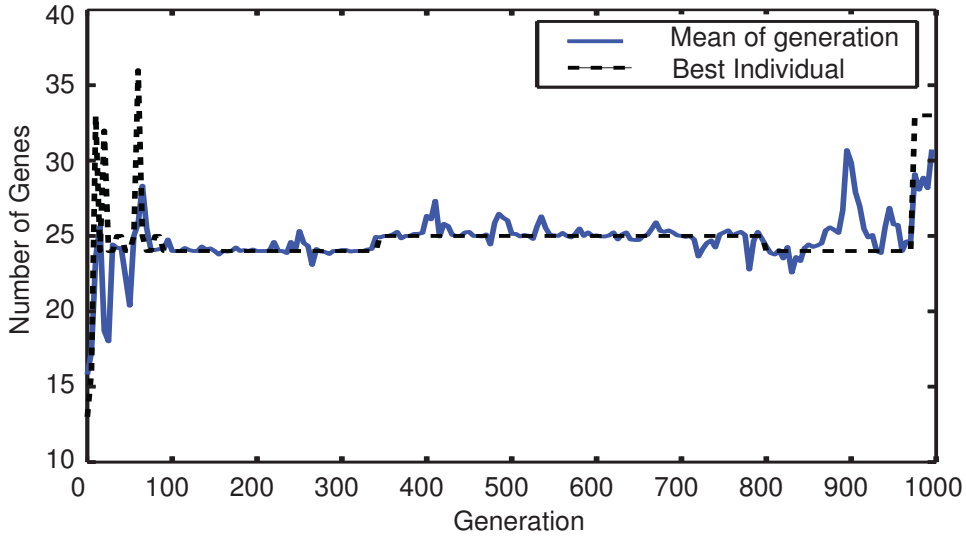


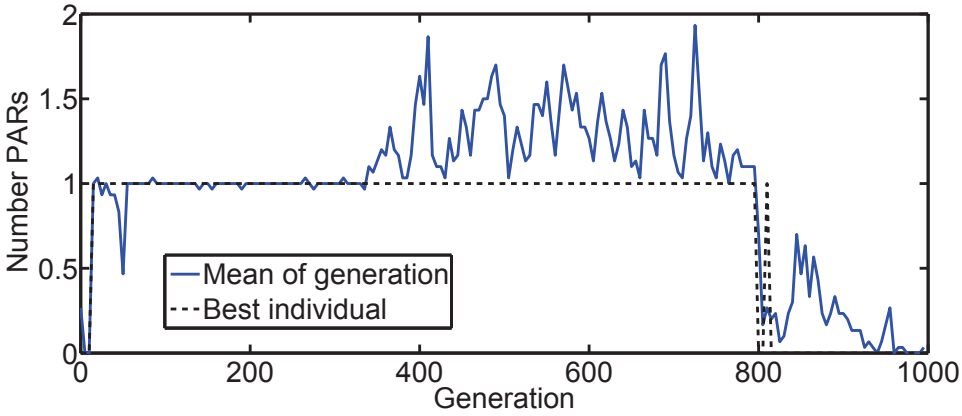
Figure 7.4: Number of genes of the best individual and the mean of the generation.

other genes, the algorithm counts more SIMs, according to the combinatorial possibilities $\binom{N}{2}$. E.g. for 4 genes the algorithm counts $\frac{4!}{2!(4-2)!} = 6$ SIMs. This masks on the one hand the number of SIMs, but on the other hand the size of the SIM is taken into account.

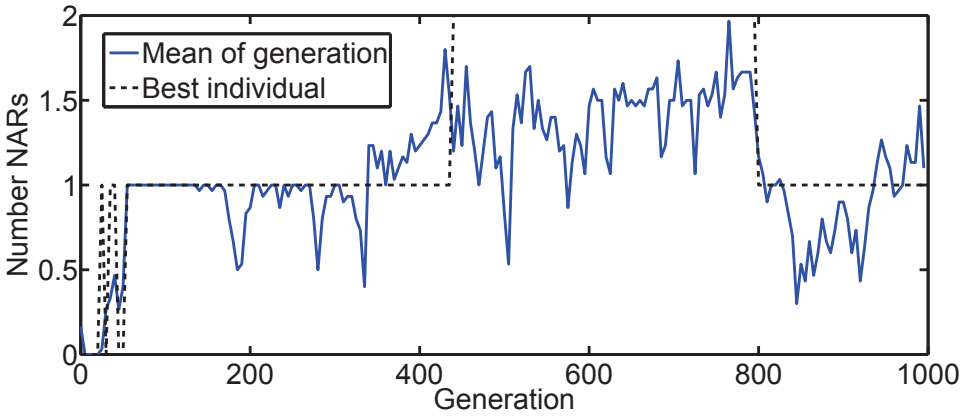
Regarding the number of most motifs, there is an increase in the beginning of the evolution and a decrease in later generations. An increase in the number of motifs is observed often between generation 300 and 500, while a considerable decrease of most motifs is observed around generation 800. The number of some motifs, e.g., I1-FFL, I1-FFL with NAR and SIM with NAR, increases again in the last generations, which can be explained by the increase of the number of genes (see Figure 7.4). The two large changes in the number of motifs correlate with two large fitness jumps. A change in the number of genes is not the reason, since the number of genes is nearly constant (see Figure 7.4). A hypothesis is, that on the one hand, evolution attempts to increase the number of motifs to perform better, whereas on the other hand, motifs that are not helpful are lost in later generations.

In the following, the change of the number of motifs is discussed in greater detail:

- **PAR**: One PAR exists in the best individual until generation 800, then the PAR is lost. On average over the generations, the number

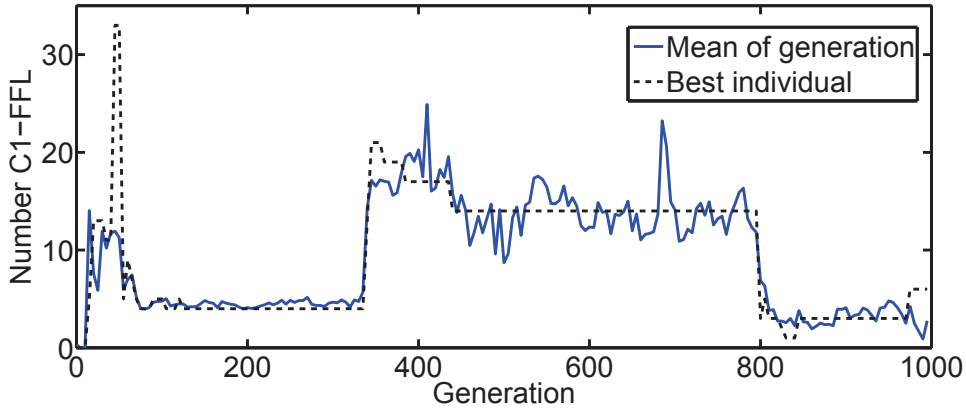


(a) Positive autoregulation (PAR)

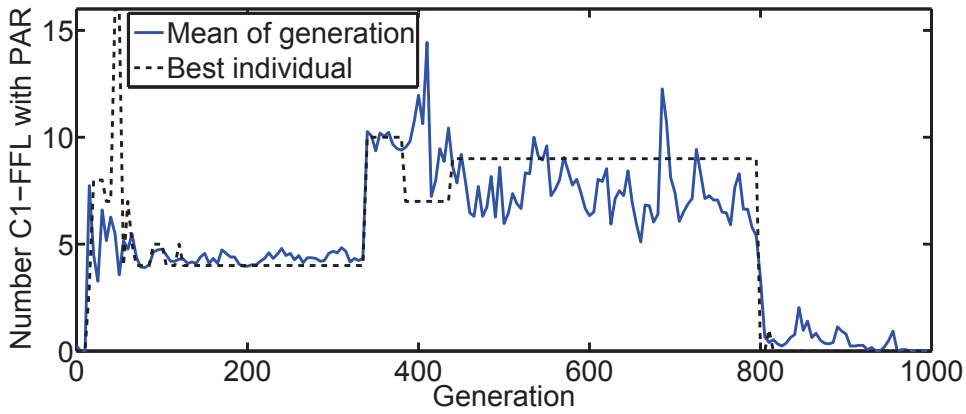


(b) Negative autoregulation (NAR)

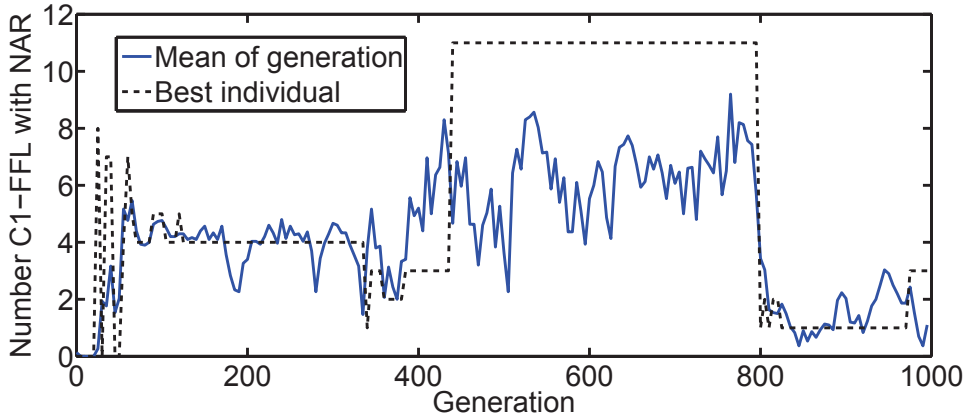
Figure 7.5: Number of autoregulations (AR).



(a) C1-FFL

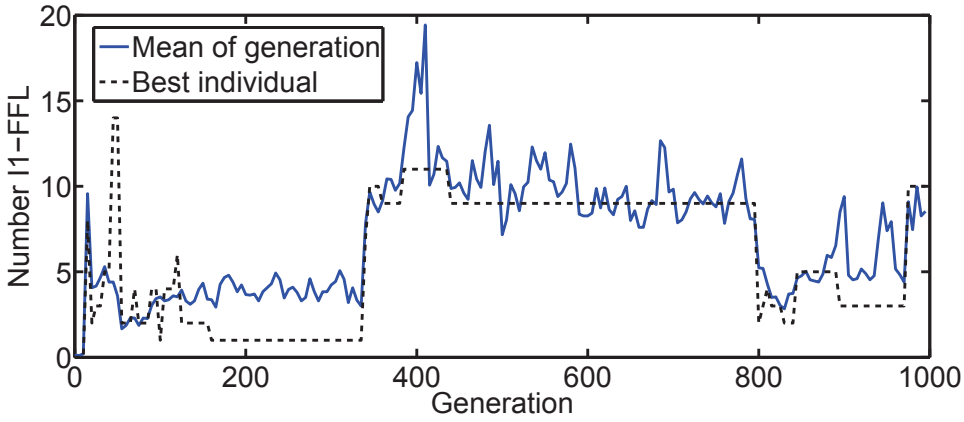


(b) C1-FFL with PAR

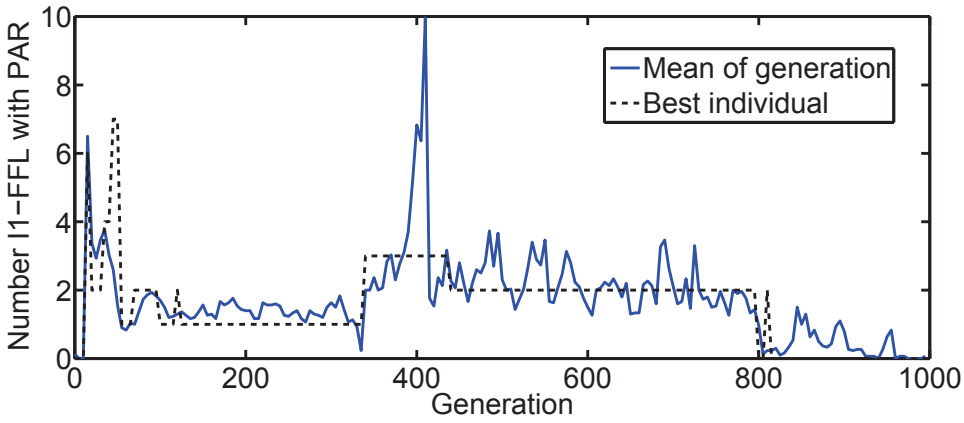


(c) C1-FFL with NAR

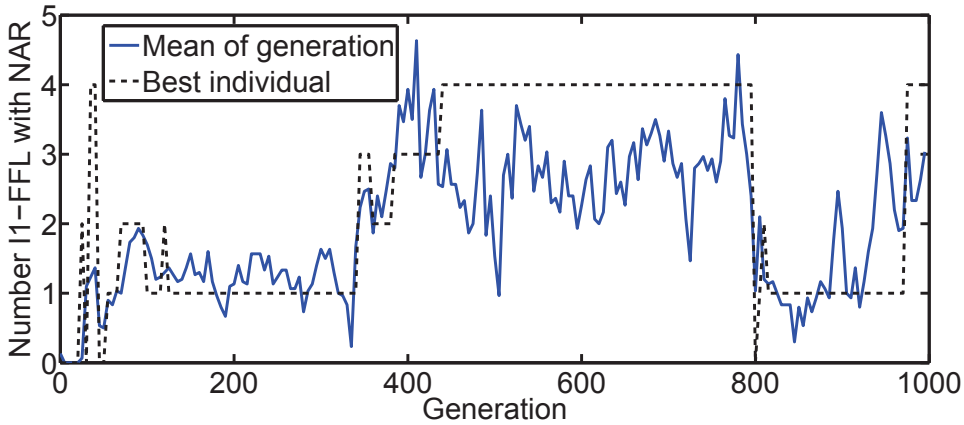
Figure 7.6: Number of coherent feed-forward loops (C1-FFL) with only activating connections.



(a) I1-FFL

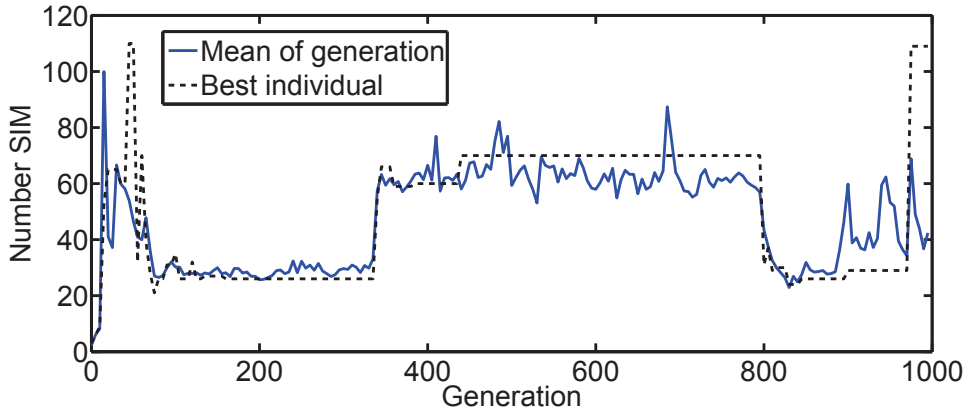


(b) I1-FFL with PAR

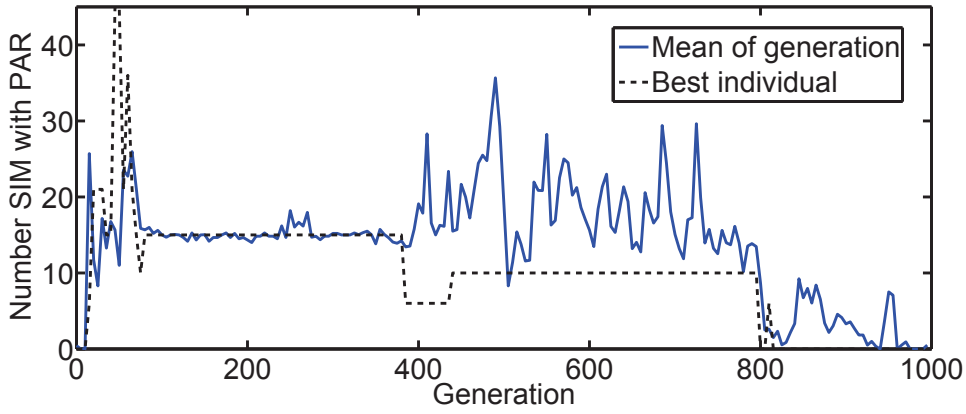


(c) I1-FFL with NAR

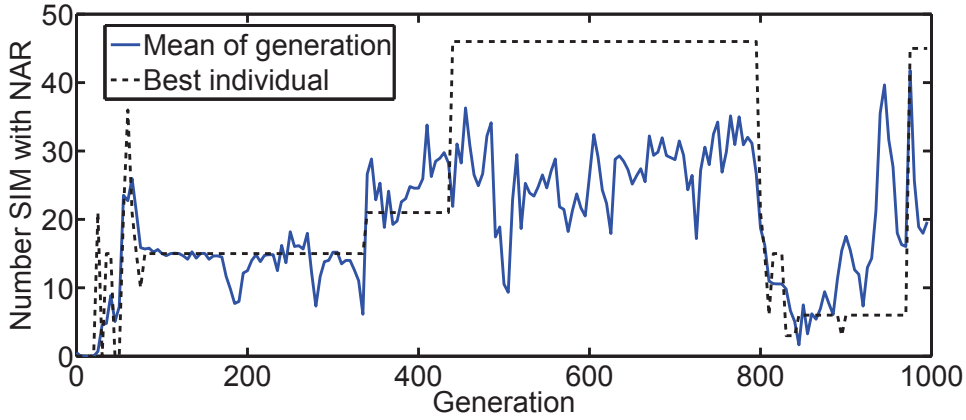
Figure 7.7: Number of incoherent feed-forward loops (I1-FFL) with one negative connection from B to C (see Figure 7.1).



(a) SIM



(b) SIM with PAR



(c) SIM with NAR

Figure 7.8: Number of single input modules (SIM) during the evolution. Three nodes SIMs are counted, so that larger SIMs result in a higher number of SIMs.

of PAR increases between generation 300 and 400 from about one to between one and two and becomes zero around generation 800.

- **NAR:** The number of NARs is very low throughout the evolution. It starts from one, goes up to two at about generation 450 and falls back to one again at generation 800.

- The number of **C1-FFL** is high during the evolution compared to that of the PARs and NARs. There is a considerable increase of this motif between generation 300 and 400 and a decrease around generation 800. The numbers of **C1-FFL with PAR** and **C1-FFL with NAR** are smaller but have a similar trend as **C1-FFL**.

- The number of **I1-FFL** is very low at the beginning and also increases between generation 300 and 400 to about 10 and decreases again around generation 800. At the end of the evolution, there is again an increase in the number of this motif. The number of **I1-FFL with PAR** and **I1-FFL with NAR** is much lower than that of the **I1-FFL**.

- The number of **SIMs** and **SIMs with NAR** is much higher than that of the other motifs. Note that we count all three-node SIMs, and consequently the larger the SIM, the more three node SIMs are counted. The change of SIMs during the evolution is comparable to that of the I1-FFL. The **SIM with PAR** is the only motif that decreases between generation 300 and 400, and reaches zero at generation 800 (because the PARs decrease to zero).

To relate the changes in the number of motifs to the effect of the genetic operators during evolution, including duplication, deletion or transposition, the ancestors of the best individual in the final generation are traced back and it is analyzed which genetic operators are selected over the generations. The results are given in Figure 7.9.

The gene deletion selected in generation 800 correlates with a strong fitness increase and a decrease of a lot of motifs. To better understand what happened during these generations, the best individual in generation 750 at the fitness plateau before the deletion and the best individual in generation 820 after the deletion are analyzed in the next section.

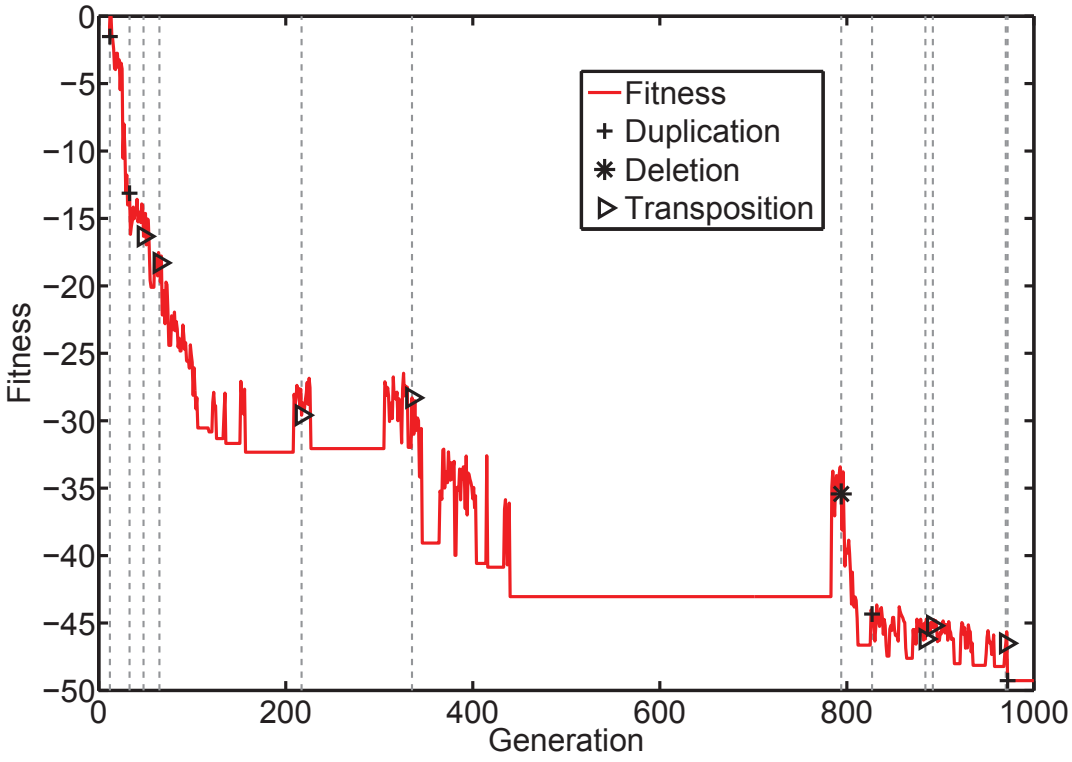
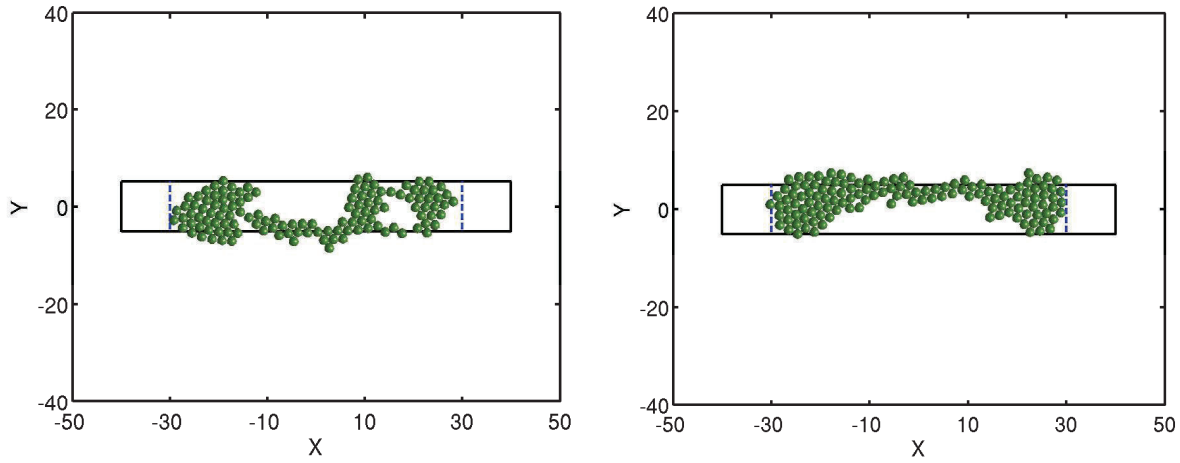


Figure 7.9: The fitness of the ancestors of the best individual in the last generation. Symbol '+' denotes a gene duplication, '*' a deletion and a triangle a gene transposition.

7.4 A Detailed Analysis of Two Individuals

The morphologies of the best individual in generation 750 (individual 750_0) and 820 (individual 820_0) are shown in Figure 7.10. The morphology of individual 820_0 is better because it is longer (the cells are closer to crossing the blue line in x-direction) and it is narrower in y-direction (the cells in individual 750_0 go further beyond the black border than those of 820_0). The genes and their activations of the two individuals are presented in Figure 7.11 and Figure 7.12. Note that only the dynamic activations are shown, and there are much more static activations.

The deleted regulatory and structural units belong to genes 9 and 10 of the best individual of generation 750. The SU for cell division of gene 9 and the complete gene 10 are deleted. Gene number 10 in the second individual is skipped to ease the comparison of the two individuals. Another difference is that the SU of gene 20 of the best individual in generation 750



(a) Individual 750_0 **(b)** Individual 820_0

Figure 7.10: Morphologies of the two analyzed individuals. The borders for fitness computation are additionally drawn to ease the comparison of the two morphologies.

changes from TF production to an unused SU through mutation. Since gene 10 of individual 750_0 has no further influence on the development (no arrows starting from this gene in Figure 7.11), the more important change seems to be the mutation of gene 20. Figure 7.13 shows the activations of the different genes in temporal hierarchies. The inhibitions are not shown and the inactivated genes are omitted. There are only temporal hierarchies and one feedback loop. The mutation to gene 20 resulted in a deletion of the whole sub-tree. The deletion of gene 9 has no further effect on the development. Gene 20 in individual 750_0 has a lot of connections to other genes and is a member of a lot of motifs. Interestingly, the loss of gene 20 resulted in an increase in fitness from generation 750 to generation 820.

7.5 Discussion

In this chapter, an analysis of the changes in the number of network motifs in the gene regulatory network during evolution of a cell growth model for an elongated body morphology is presented. A general trend is that the overall number of motifs increases significantly at the beginning of evolution. During the evolutionary process the numbers of all motifs have increased except for PAR.

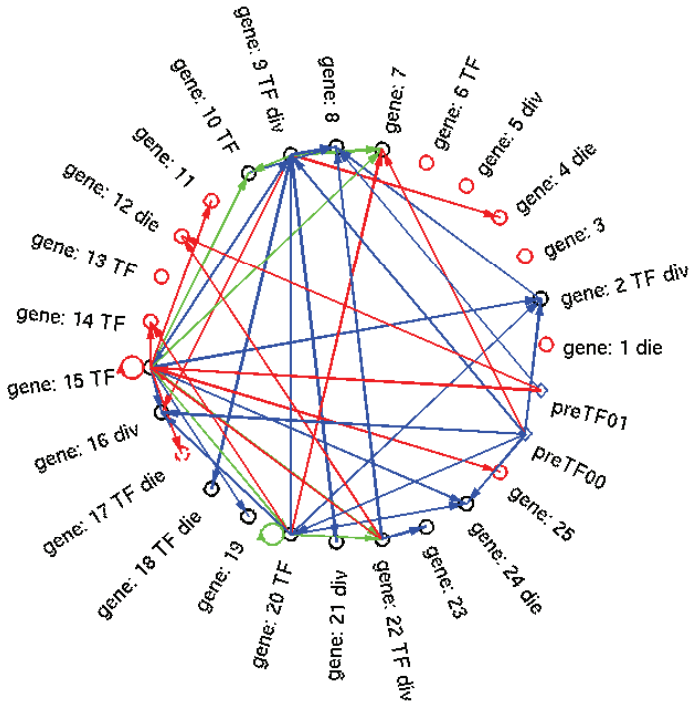


Figure 7.11: The genes and their used connections of the best individual in generation 750. The circles represent the different genes. Genes that are active during development are denoted with black circles. Red circles indicate genes that are never active. The arrows represent the interactions between the genes, where blue represents an activating, red an inhibiting and green both an activating and inhibiting connection. The two diamonds represent the predefined TFs.

Since the genome length does not change significantly during evolution, it seems that it is not just the increase of genetic material but of **structured genetic material**, i.e., dynamic network motifs, that is important during the evolutionary process. At the same time, motifs that do not influence development are lost again during evolution. Therefore, it seems that the occurrence of motifs is under selectional control and that the increase of dynamic network motifs is related to the evolvability of the process.

The genetic changes that contributed to the fitness jump around generation 800 are analyzed and therefore, the genes of two individuals before and after the genetic change are compared. The fitness increase and the decrease of the number of dynamic motifs are caused by one mutation that changed a gene from producing an important TF to a gene without func-

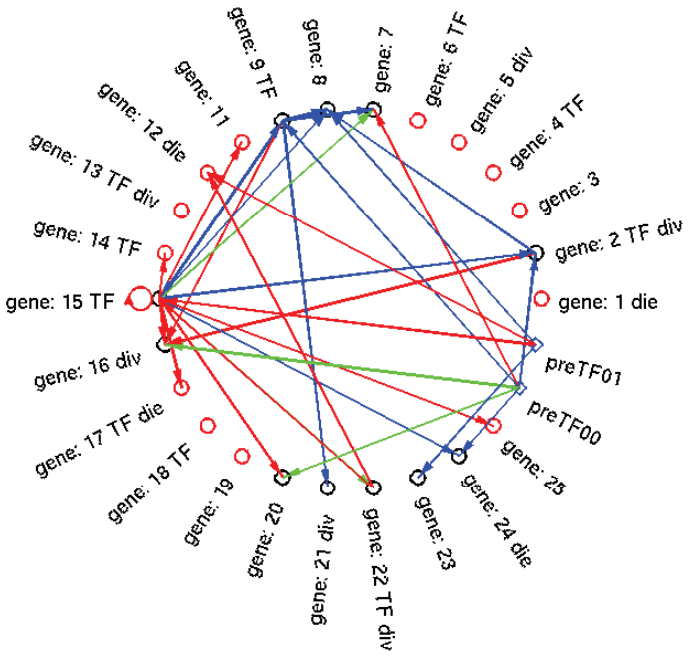
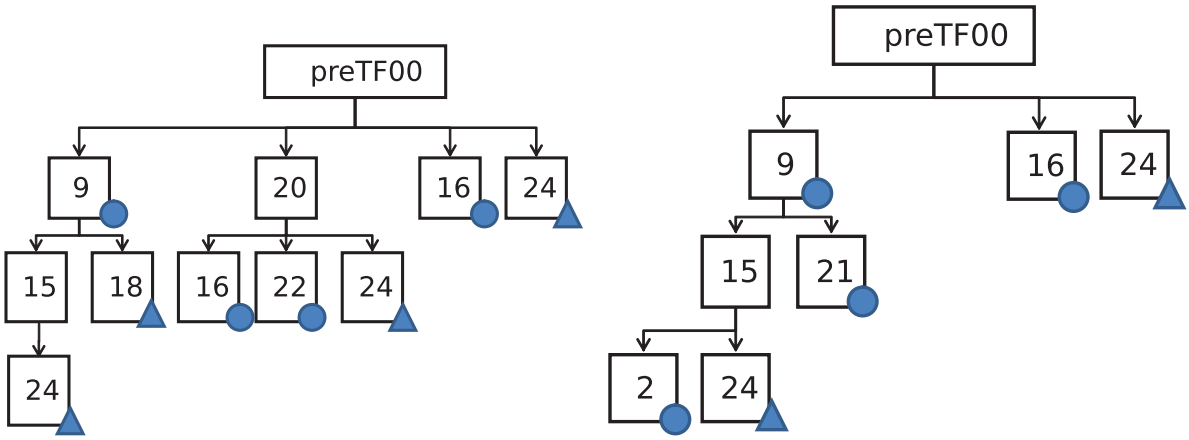


Figure 7.12: The genes and their used connections of the best individual 820. Notation as in Figure 7.11. The genes are numbered, and number 10 is skipped for an easier comparison between the two individuals of generation 750 and 820, because gene 10 was deleted in between.

tion. Contrary to intuition, the correlated gene deletion neither influenced the fitness nor the number of motifs.

A more detailed interpretation of the results is restricted by the fact that only observations from one experiment are available. Needless to say that a more statistically sound analysis would be desirable, however, the considerable computational expense of the described process makes it difficult to run a larger number of experiments.

For the analysis of static motifs, other authors have normalized their results to the motifs one can find in random networks. For dynamic motifs this is difficult, because most static motifs in random networks will not be dynamical, simply because the developmental process terminates very early. Frequently, this is due to the early activation of cell death by a prediffused TF in random networks. More precisely, most random networks result in an activation of cell death in the first developmental step and the development stops. This results in nearly no network motifs, because no TFs are produced. In order to make sure that during the evolu-



(a) Individual 750_0 **(b)** Individual 820_0

Figure 7.13: The activating relations of the different genes. Genes for cell division are marked with a circle, genes for cell death with a triangle. Only some important activating effects are shown, inactivated genes and inhibiting connections are omitted.

tionary process not just the raw genetic material is increased we compared the number of dynamic motifs to the genome length during evolution.

In the next chapter, I present an analysis of the redundancy in GRNs during the evolution.

8 Redundancy in the Evolution of Gene Regulatory Networks

The previous two chapters focused on the developmental behavior, namely stability and repair, and their genetic causes, the network connections and motifs. In this chapter I investigate the influence of redundancy on the evolutionary performance of a GRN, since redundancy is believed to play a key role in the robustness and evolvability of biological systems.

Extensive simulation results suggest that, for the developmental model studied in this work, maintaining sufficient redundancy helps to improve the ability of the evolutionary algorithm to achieve better performance. To examine the change of redundancy during the evolutionary process and its relationship to evolutionary performance in more detail, a quantitative definition for measuring different aspects of redundancy, namely, structural redundancy, functional redundancy and functional proximity are proposed. The results show that the evolution attempts to increase the functional redundancy after pruning of redundant genes if the evolution is operating under a larger selection pressure. It is also interesting to notice that an increase in functional proximity is positively related to an increase in evolutionary performance.

The following section contains a general definition of redundancy and robustness and provides the definition of a few concepts related to redundancy that will be analyzed in this chapter. The definitions are inspired by biological observations and adopted to fit the computational framework. The evolutionary algorithm used for evolving the developmental model, the fitness function of the evolution and a number of experimental setups for studying the role of redundancy are provided in Section 8.2 and are based on the experiments in Chapter 5. Results from the different experimental setups are presented in Section 8.3, analyzes on different redundancy measures are provided in Section 8.4 and Section 8.5. A discussion is given in Section 8.6. Parts of this chapter are based on [116].

8.1 Redundancy during the Evolutionary Process

Systems biology aims at understanding biology with regards to four systems properties, namely, system structure, system dynamics, control methods and design methods [77]. Among others, robustness is one of the most important design principles of biological systems.

Biological robustness can be achieved by a variety of mechanisms [77]. It has been found in [137] that genetic redundancy is one of the main mechanisms in biology that contribute substantially to mutational robustness. Meanwhile, an inherent trade-off between redundancy and evolvability has been revealed in [65] for a redundant genotype-phenotype mapping. Such a trade-off is confirmed from a slightly different perspective in [139] by presenting the evidence that degeneracy, a partial redundancy, is a fundamental source of both robustness and evolvability.

A concept that is closely related to redundancy and robustness is neutrality [134]. Kimura has been the first to notice the role of neutrality in biological evolution [74]. He has argued that most allelic variation and substitution is neutral but a lot of mutations are deleterious. This suggests that random genetic drift may be one of the main driving forces behind evolution. The relationship between neutrality and robustness has also been widely studied in evolutionary computation [37]. Yu and Miller [143] analyzed different problems with different types of neutrality and found that redundancy can but need not be beneficial for evolution depending on the implementation. Banzhaf [10] proposed a model using a genotype-phenotype mapping with neutrality and found that neutrality allows the system to work more flexibly.

The aim of this chapter is to achieve a better understanding of the ways in which the evolutionary process succeeds in building regulatory systems. In particular, it should shed some light on the role of redundancy during this process by analyzing the evolution of simple models of regulatory systems in computational simulations.

Numerous definitions for redundancy have been proposed in the literature both in an engineering as well as a biological context. Here, the redundancy during the evolutionary design process as opposed to during the operation time or lifetime are analyzed. In designing engineering systems using direct redundancy, we usually duplicate system components to increase robustness and fault-tolerance, i.e., the additional components are only active once the working components fail. These components are

redundant in the sense of 'not being used' during normal operation and usually do not play a key role during design. They are most likely added to the system after the major design phases have been concluded. In biology, gene duplication plays a very important role during evolution for acquiring new genetic raw materials that can potentially lead to evolutionary innovation [142]. In the first step, gene duplication leads to *genetic redundancy*, because two segments of genes now encode the same functionality. Therefore, it is reasonable to say that genetic redundancy possibly constitutes the first step toward evolutionary innovation. In biology, genetic redundancy resulting from gene duplications has four possible fates: (a) neo-functionalization, i.e., genes assume a new functionality which is preserved by natural selection; (b) non-functionalization, i.e., genes become pseudogenes, (c) sub-functionalization, i.e., duplicates of a gene with multiple functions carry reduced, complementary sets of functions, and (d) the original and the duplicated genes assume overlapping functionalities. Recently, it has been suggested that bacteria can contain a substantial number of pseudogenes for a limited period of time [1]. Therefore, it seems that genetic redundancy has a limited time window within which it can be turned into evolutionary innovation. Lynch and Connery estimated the average time window for a gene duplication to be about 4 million years [91].

So far we have mainly focused on redundancy as a necessity for the evolutionary process to have genetic materials that can assume new functions, i.e., evolutionary innovations. However, redundancy has also been believed to be a means for providing organisms with mutational robustness in particular for small population sizes [85]. As mentioned, it becomes evident that redundancy plays different roles during the evolutionary process.

In order to get a better understanding of these different roles, three different measures related to redundancy, which are tuned toward the influence of redundancy during the design phase, are introduced in the following.

- **Structural Redundancy**
  Structural redundancy R_S denotes the disconnectedness or the non-functionality of a gene. In this notation, pseudogenes would be structurally redundant:

$$R_S = \frac{N_{R_S}}{N}, \tag{8.1}$$

where N_{R_S} denotes the number of structurally redundant genes in the whole genome containing N genes.

- **Functional Redundancy**
  Functionally redundant genes are those whose deletion would have no effect on the phenotype. Thus structurally redundant genes are also functionally redundant but not necessarily vice-versa. E.g. genes can express certain proteins, which, however, have no or negligible effect on the phenotype. In this notation, most gene duplications lead to functional redundancy:

$$R_F = \frac{N_{R_F}}{N}, \quad R_F \geq R_S, \tag{8.2}$$

  where N_{R_F} denotes the number of functionally redundant genes in the whole genome containing N genes.

- **Functional Proximity**
  All redundancy measures are generally discrete: a gene is either functionally redundant or not. However, for the evolutionary process it is interesting to analyze how far away redundant genes are from assuming functionality, e.g., on average how many mutations or how much time is needed in order for evolution to either turn the genetic redundancy into innovation or to get rid of the genetic materials that have turned into pseudogenes. *Functional Proximity* measures the transition probability from a functionally redundant gene to a functional gene under mutations. If we denote functional genes by g and functionally redundant genes by $g_{R_F} \in G_{R_F}$, then we can define the transition probability $P_\Theta(g_{R_F} \to g)$ through a mutation parametrized by Θ and *Functional Proximity FP* by

$$fp = P_\Theta(g_{R_F} \to g) \tag{8.3}$$

$$FP = \frac{1}{N_{R_F}} \sum_{g_{R_F} \in G_{R_F}} P_\Theta(g_{R_F} \to g). \tag{8.4}$$

Table 8.1: Definitions of the different setups

Setup no.	Specification
setup 1	never prune (same setup as in Chapter 5)
setup 2	prune in generation 500
setup 3	prune every 100th generation
setup 4	prune every 10th generation
setup 5	prune once, when fitness of best individual crosses -40
setup 6	fixed DNA with mutation, without duplication, deletion and transposition using 24 RUs and 8 SUs. The order of the RUs and SUs is predefined, also the type of the SUs.
setup 7, 8, 9, 10	fixed DNA with mutation and transposition and without duplication and deletion. The number of RUs and SUs is 30, 50, 100, 500 respectively.

8.2 Experimental Setup with Varying Redundancy

The experimental setup is based on the fundamental experiments in Chapter 5. This setup is compared to other setups, where all functionally redundant genes found in the chromosome are pruned. A gene is considered as functionally redundant if the deletion of the gene results in no fitness change. It should be pointed out that pruning of functionally redundant genes is different to gene deletion in that deletion of a randomly chosen sequence of RUs and SUs may change the fitness of the individual.

To investigate the influence of redundancy on the performance of evolution, 10 different pruning setups for comparison are examined. The definitions of the different setups are listed in Table 8.1. 15 evolutionary runs with different random seeds for each setup are performed.

Setup 6 is designed for investigating the performance of evolution if compact chromosomes are used. In this setup, the positions of all RUs and SUs and the types of the SUs are predefined. The predefined genome is shown in Figure 8.1, the structure of one individual that achieved the optimal fitness obtained in this setup is provided in Figure 8.2.

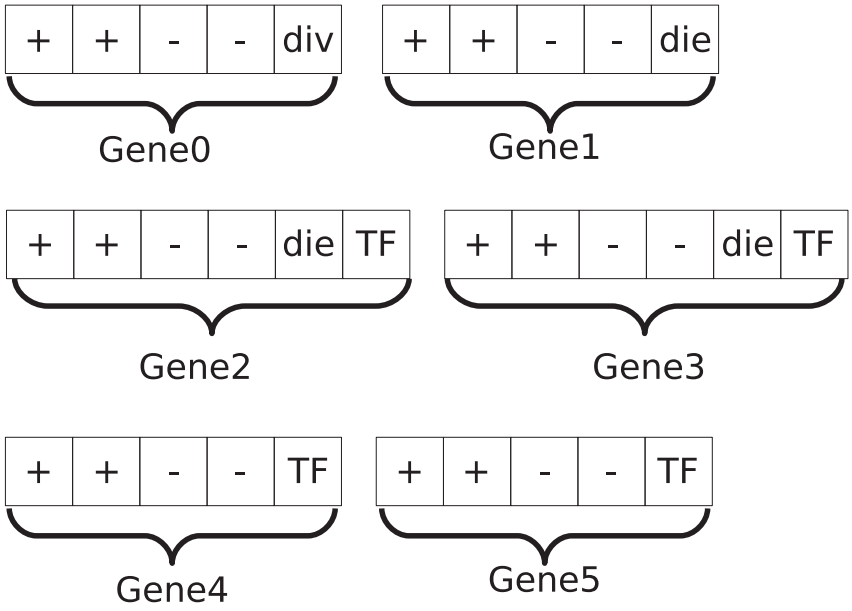


Figure 8.1: A predefined chromosome in setup 6, where the positions of all RUs and SUs, the sign of the RUs and the type of the SUs are fixed.

8.3 Evolutionary Results

The boxplots of the best fitnesses from 15 independent runs for the first 9 setups are given in Figure 8.3. Note, however, that in setup 10, all 15 runs result in a fitness of 600, which means there are no cells at the end of the development. Therefore, the results are excluded from the Figure. The detailed fitness profiles for setups 1, 4, and 7 are shown in Figure 5.3, Figure 8.4, and Figure 8.5. The other results are in Appendix B.

The differences of the means are tested with the Mann-Whitney U test with a statistical significance of 95% (see [50]). The means of setup 1, 2, 5, 8, and 9 are lower than the ones of setup 4 and 6. Additionally setups 3 and 7 are better than setup 6, setup 2 and 8 are better than setup 3. More experiments would be helpful to increase the statistical significance, e.g. the difference in the length of the 25th and 75th percentiles of setup 4, 6 and 8 should become smaller.

The results of setups 1 to 5 suggest that more frequent pruning leads to a worse performance. Note, that setups 1, 2 and 5 perform comparably well, which suggests that pruning of redundant genes in a late stage of evolution, or when the evolution is already more or less close to the optimal solution, will not degrade the evolutionary performance. Basically, this means that no genetic ‘raw material’ is needed anywhere in later generations.

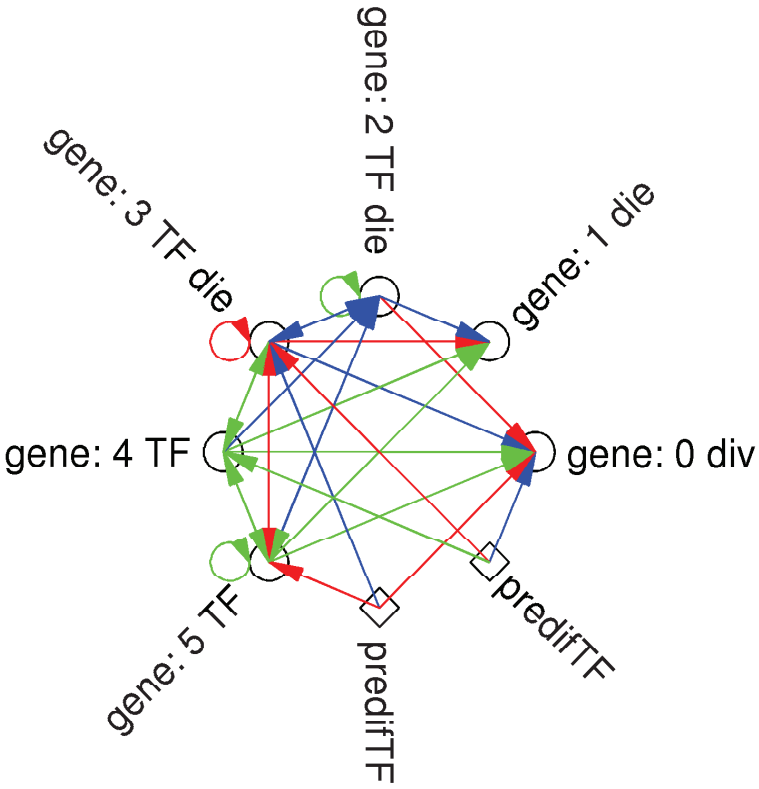


Figure 8.2: The genome and its connections of a good individual (the fitness is optimal) of setup 6. The dots are the genes, the predefined TFs are diamond shaped. The arrows define the activations between the different genes, an activation is represented by a blue line, an inhibition by a red line and the green lines are both, activations and inhibitions.

On the other hand, the results from setup 3 (pruning every 100th generation), which are worse than those from setups 1, 2 and 5 (yet not statistically significant), indicate that more frequent pruning tends to worsen the performance of the EA. The results of setup 4 (pruning every 10th generation), which are significantly worse than those in setups 1, 2 and 5, confirm that continuous pruning of the redundant genes leads to much worse performance.

It is worth discussing the results of setup 6. Although this setup has the fewest parameters to be optimized, only one of the 15 runs converges to the optimal fitness. This indicates that for a representation that does not allow functional redundancy, the evolution has a big difficulty in finding the optimal solution, even if the optimal solution exists (as shown in Figure

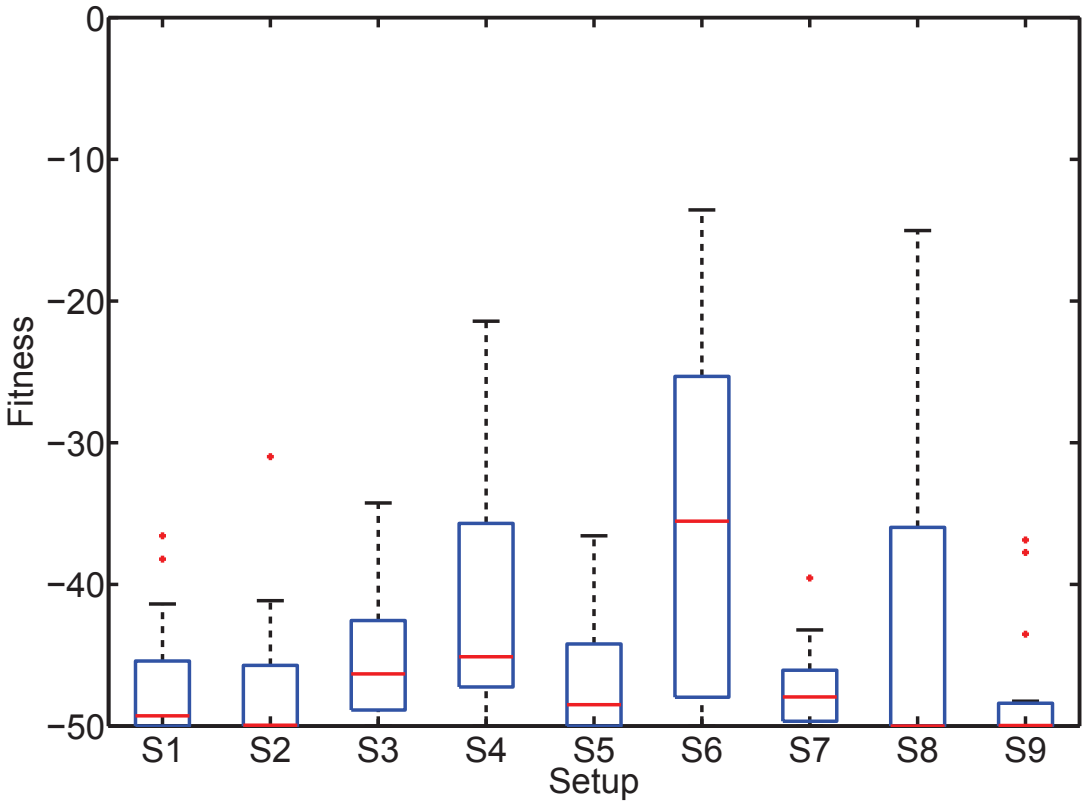


Figure 8.3: The boxplots of the best fitness from 15 independent runs of setup 1 to 9.

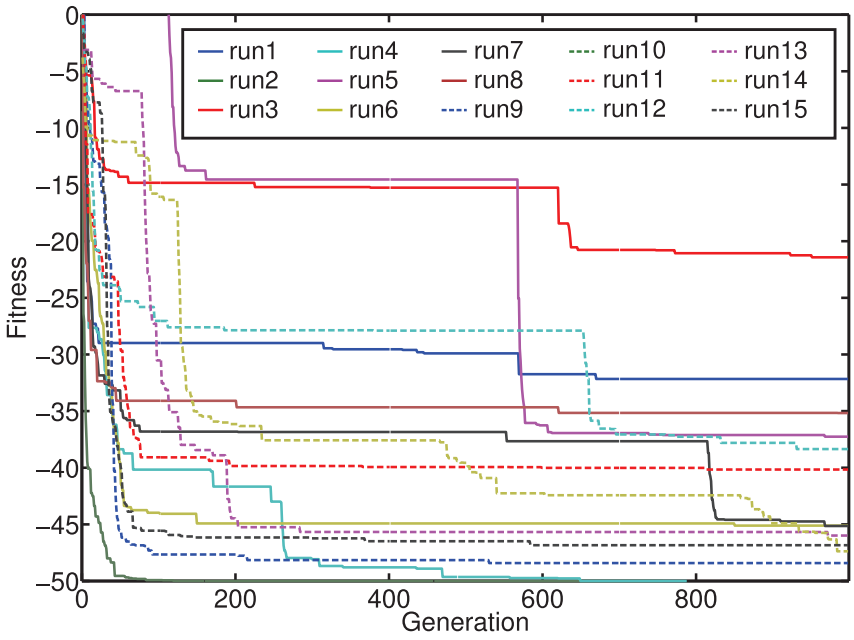


Figure 8.4: Fitness curves of setup 4. The fitness of setup 4 run 10 is always 600 and not displayed here.

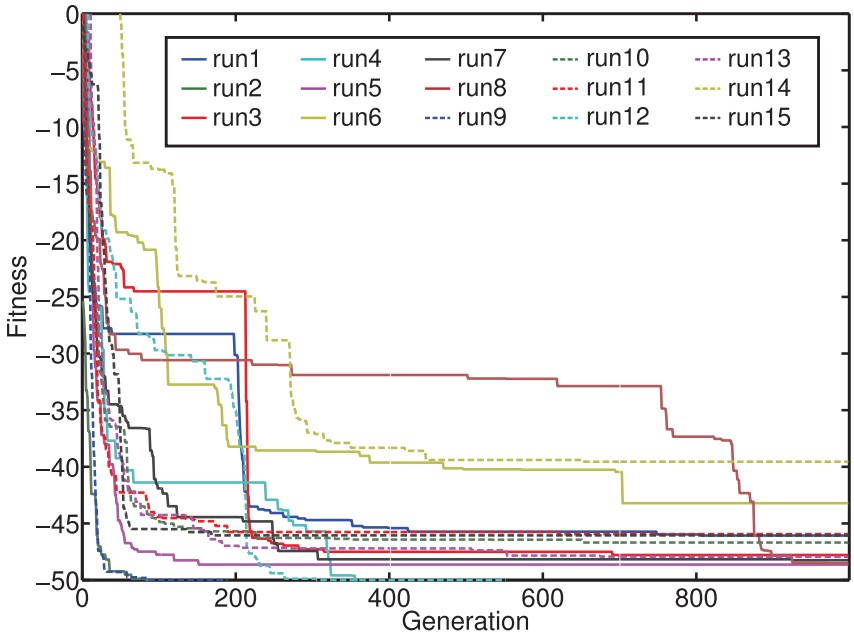


Figure 8.5: Fitness curves of setup 7.

8.2). This result also supports the hypothesis that redundancy improves evolvability.

A common belief in evolutionary computation, where direct coding is often used, is that the performance of evolutionary algorithms does not scale well with the search dimension. The results from setup 9 show surprisingly that this belief might not be correct for developmental systems. However, it should be noted that the extremely poor results in setup 10 (500 RUs and SUs), in which none of the runs have been successful, indicate that there is a certain upper bound of the search space in which evolution can no longer work properly.

To examine the possible correlation between the genetic variations and the change in fitness, the evolutionary 'history' of the best individual in the last generation of run 1 and run 5 of setup 1 are analyzed. Figure 8.6 and Figure 8.7 show the fitnesses of the best individual of each generation and of the ancestors of the best individual of the last generation, together with the number of RUs and SUs in the individual. In run 1, refer to Figure 8.6, the increase in fitness at about generation 600 correlates with a change in the number of RUs and SUs, where a gene deletion is performed. In run 5, refer to Figure 8.7, a few 'jumps' in fitness correlate to a gene duplication or deletion (in generations 800 and 950). However, the increase in fitness

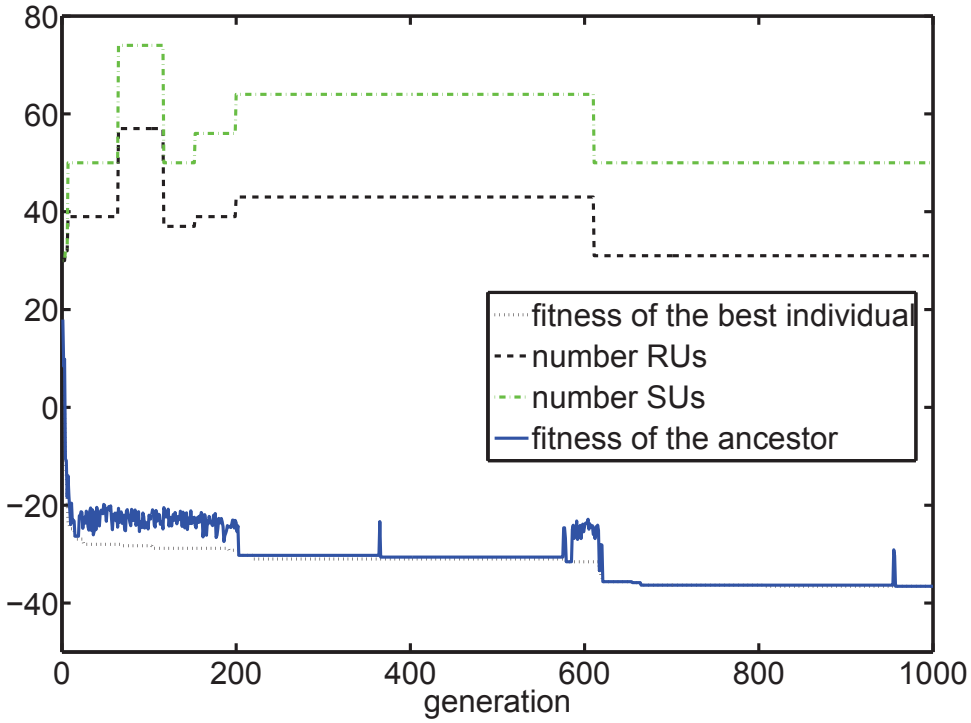


Figure 8.6: Number of RUs and SUs and fitness of the ancestors of the best individual from setup 1, run 1.

between generations 300 and 400 has not resulted from duplication or deletion. In other words, this change in fitness may be a result of gene transposition or mutation. Note, however, that the ancestor of the best individual is often not the best individual before a 'jump' in fitness.

8.4 Analysis of Structural and Functional Redundancy

To quantitatively analyze the role of redundancy in evolution, the structural and functional redundancy defined in Equation 8.1 and Equation 8.2 based on the results from the first 10 runs of setup 1 are computed. The number of structurally redundant genes N_{R_S} counts the genes that can never be activated or have no functionality, e.g., those genes have no SUs for TFs, cell division or cell death. Genes that are functionally redundant are detected by comparing the fitness values with and without a gene. If the deletion of the gene does not influence the fitness, then the gene is functionally redundant. Both structural and functional redundancy for

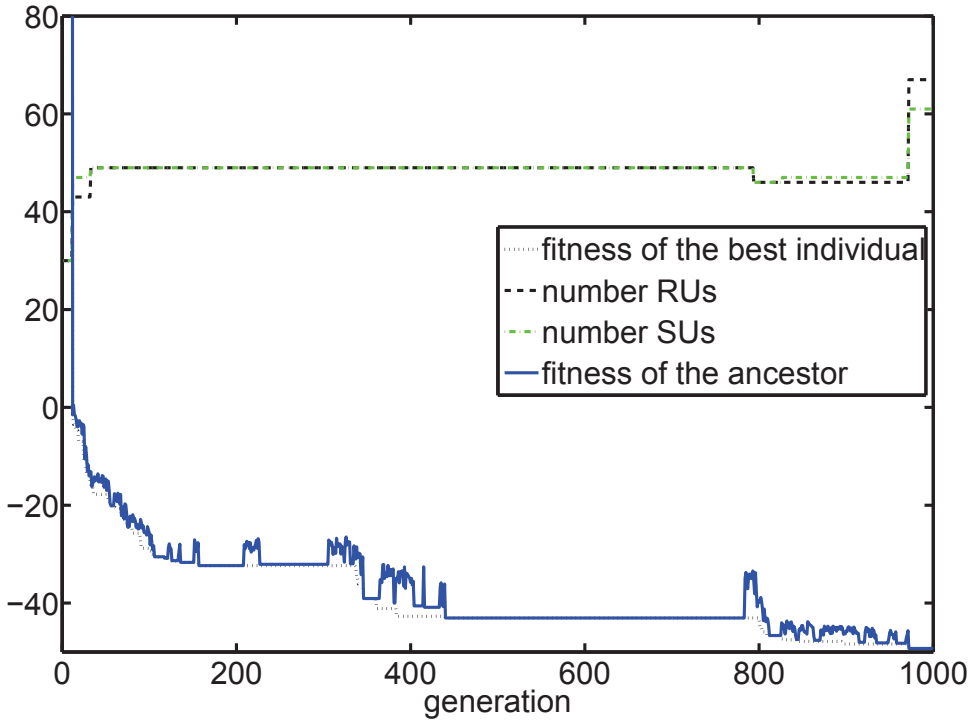


Figure 8.7: Number of RUs and SUs and fitness of the ancestors of the best individual from setup 1 run 5.

the ancestors of the best individual in every 10th generation are computed, see Figure 8.8. From this Figure, no clear tendency in the profile of the redundancy measures can be observed. One conservative observation we can make is that values of R_S and R_F are within a certain range: $20\% \leq R_S \leq 40\%$ and $70\% \leq R_F \leq 90\%$, with a few exceptions. There is also no significant difference in the profiles of the redundancy measures between the successful and the nonsuccessful runs. Given the hypotheses that redundancy improves evolvability, an expectation was an increase in functional redundancy during evolution, which is not confirmed by the results shown in Figure 8.8. One explanation might be that the functional redundancy of the individuals in this run is already very high from the beginning, which is sufficient for creating innovative solutions. Thus, a higher value of functional redundancy is no longer necessary.

In order to test this hypothesis, additional experiments, where redundant genes are pruned in generation 100, are performed. Then the structural and functional redundancy for the remaining generations are computed. Figures 8.9 and 8.10 present the number of redundant genes during the evolution and the fitness values of the related individuals. From these

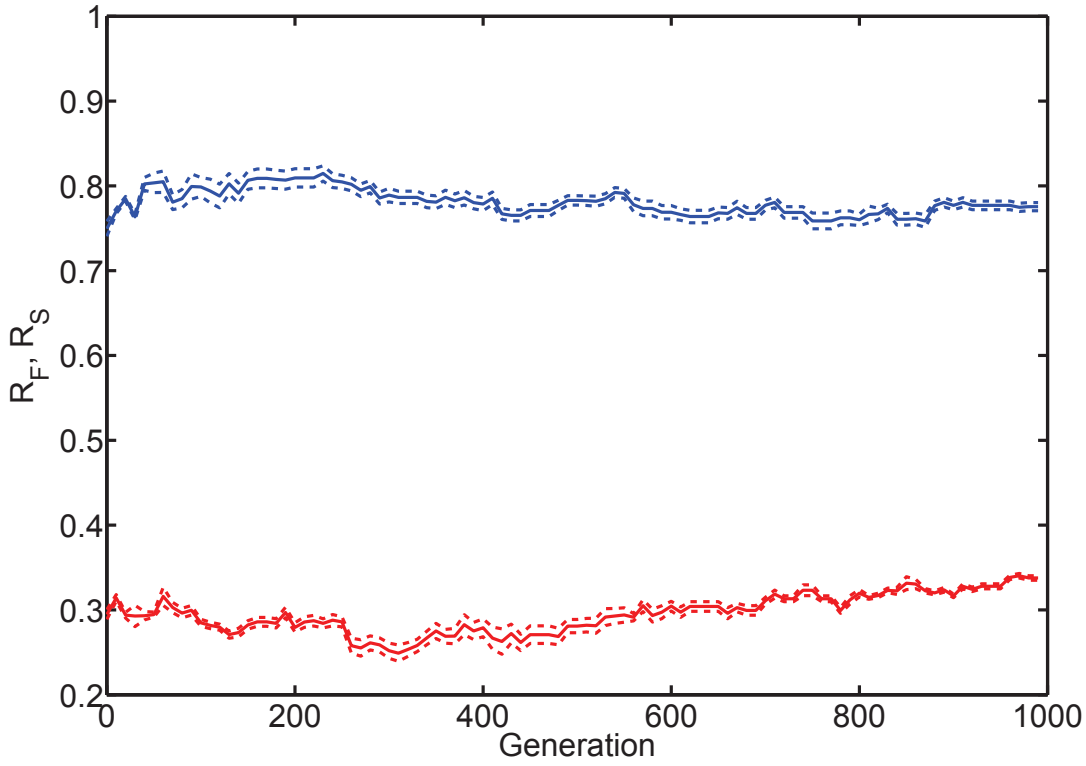
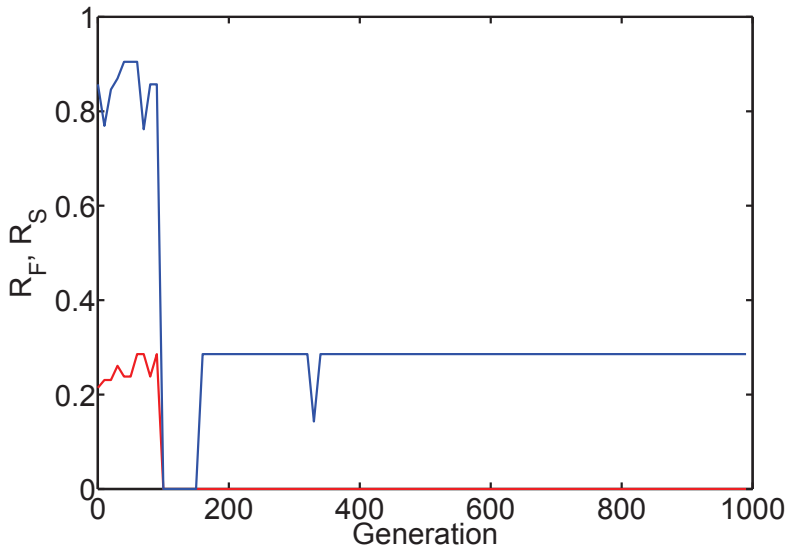
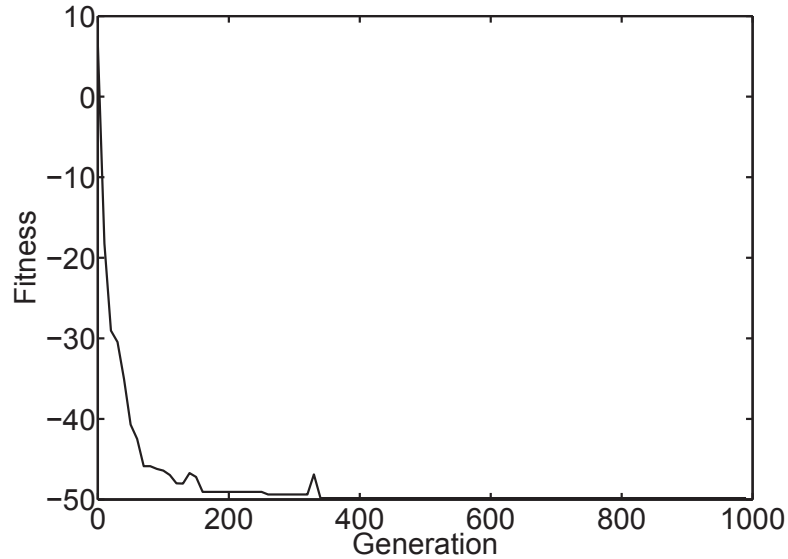


Figure 8.8: Mean value and variance of R_S (red) and R_F (blue) of the first ten runs of setup 1 ($R_F > R_S$).

Figures, we see that the first redundant genes appear between generations 50 and 100 after the pruning. In the first run, the fitness converges very fast to the optimum. Meanwhile, we see that the structural and functional redundancy keep constant after generation 350. This is reasonable since the best solution has already been found and there is no selection pressure for a better fitness. In this case, additional redundancy is not needed. In the second run, in contrast to the results in the first run, the functional redundancy first rises to 30% and then increases further to nearly 70% which is in the range of Figure 8.8. This implies that the evolution is under pressure to increase redundancy since the quality of the population is still low (respectively the fitness is high). Note, however, that the statistical significance of the above observations remains to be verified, since they are made only based on two individual runs.

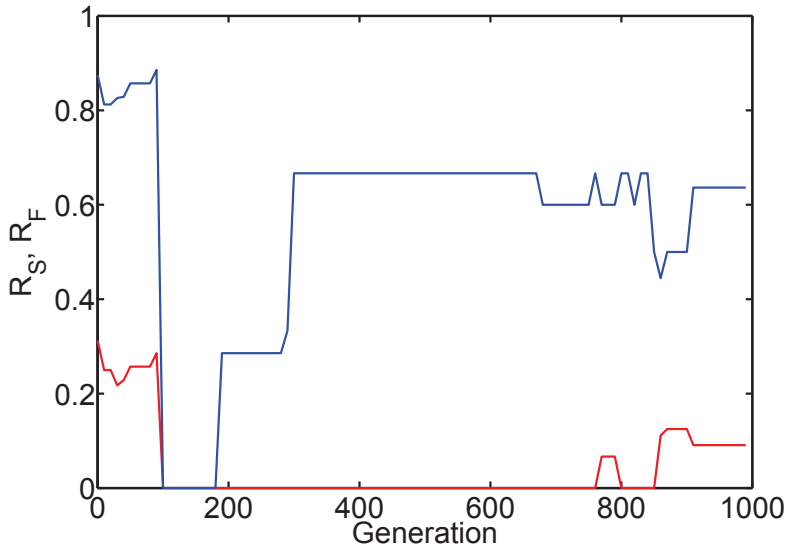


(a) R_S (red) and R_F (blue), $R_S \leq R_F$.

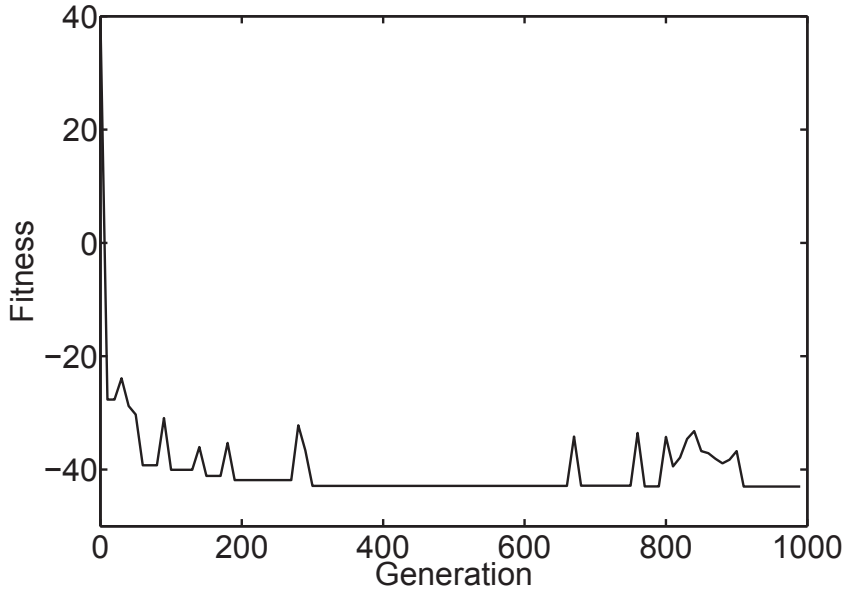


(b) Fitness curve

Figure 8.9: R_S, R_F and fitness curve of run 1 after pruning in generation 100.



(a) R_S (red) and R_F (blue), $R_S \leq R_F$.



(b) Fitness curve

Figure 8.10: R_S, R_F and fitness curve of run 2 after pruning in generation 100.

8.5 Analysis of Functional Proximity

In the following, the functional proximity (FP) of different individuals taken from different stages of the evolutionary process are analyzed. The best individual from the beginning, the middle and the end of each evolution is taken (generation 100, 500 and 999). For this purpose, the individuals are mutated 200 times and their functional redundancy R_F is computed. The functional proximity of gene j can be estimated by

$$\hat{f}_{p,j} = \frac{1}{N_{mut}} \sum_{i=0}^{N_{mut}} R_{F,i,j}, \tag{8.5}$$

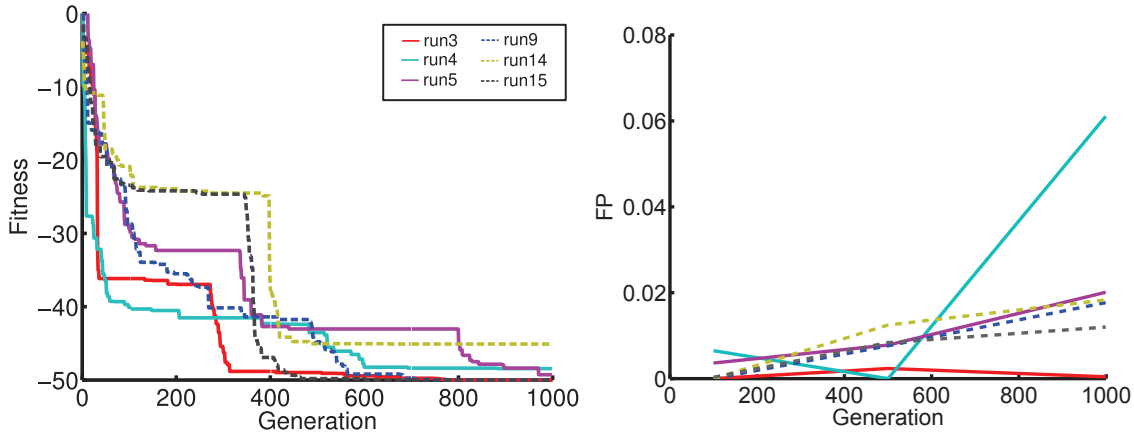
where the number of mutations $N_{mut} = 200$. The functional proximity $\widehat{FP}$ of the individual is estimated by

$$\widehat{FP} = \frac{1}{N} \sum_{j=0}^{N} \hat{f}_{p,j}, \tag{8.6}$$

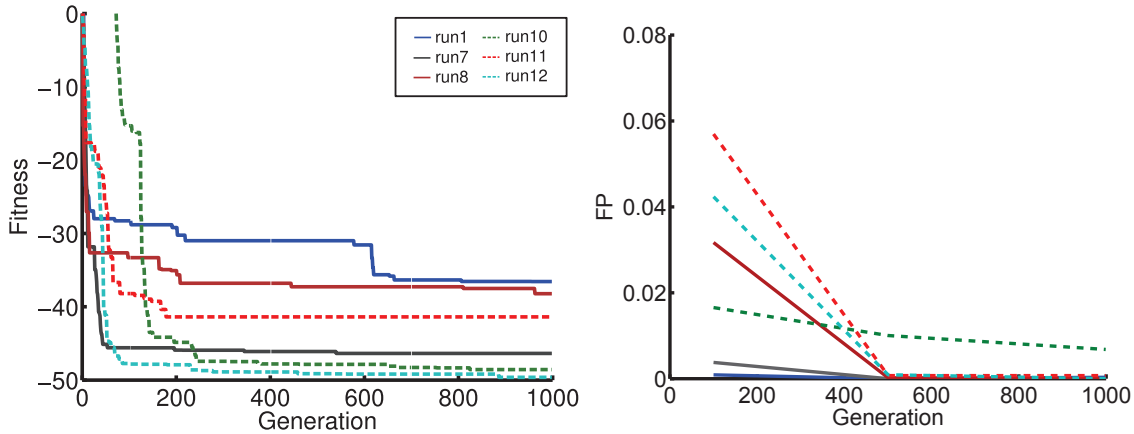
where N is the number of genes of the individual.

Figure 8.11 shows the fitness curves and the associated FP. To analyze the FP, the different evolutionary runs are separated into different categories. The first category (see Figure 8.11a) comprises fitness curves with a step by step increasing fitness and good results at the end. The curves assigned to the second category (see Figure 8.11b) show few and small fitness jumps or bad results. The huge fitness increase at the beginning is not taken into account. The three runs of the third category (see Figure 8.11c) found the global optimum very fast, so the influence and trend of the FP cannot be analyzed.

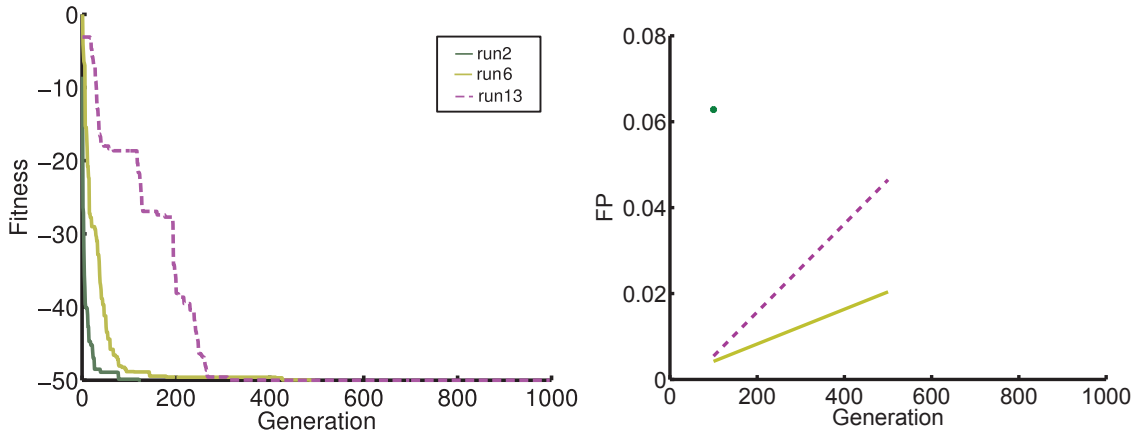
In general, few functionally redundant genes change to a functional gene. The FP of the curves of the first category increases in four of six cases. In the second category, the FP decreases to very small values at generations 500 and 999 in five of the six cases. The optimizations in the third category found the global optimum fast, so it is difficult to analyze a tendency, but the value of FP is high or increases to a high value. This supports the assumption that high, increasing values of FP are beneficial for evolution. The values at generation 100 seem less significant, because the runs in the second category have different values of FP at generation 100 and there is no clear relation to their performance, e.g. run 12 (Figure 8.11b)



(a) The evolutionary runs shown in these figures show an increasing fitness with good results at the end.



(b) These runs converged fast or have bad fitness values.



(c) The runs shown here found the global optimum too fast to analyze the influence of the functional proximity.

Figure 8.11: The functional proximity is shown on the right side and the associated fitness curves are on the left. The results are assigned to three different categories.

has a high FP and shows good performance at the beginning, but run 11 (Figure 8.11b) has a higher FP but the performance is worse.

An exception is run 3 (Figure 8.11a), that has a small value of FP in all three test generations but shows good results. The FP of run 4 (Figure 8.11a) shows no clear tendency, it first decreases to a very small value and then increases to a very high value and its performance is good. Run 10 (Figure 8.11b) shows low performance and a high but decreasing FP. Recall that an evolutionary optimization is a stochastic process and the FP is a probabilistic value. Nevertheless, these results suggest that there is a positive correlation between an increase of functional proximity and a performance increase of the evolution.

8.6 Discussion

In this chapter, I analyzed the role of redundancy during evolution in a simplified computational model for the development of a cellular elongated artificial organism with GRNs.

In a first set of experiments, the redundancy of the different genomes was limited by pruning all functionally redundant genes in a variety of setups. Statistical results show that there is a significant decrease in the performance of the evolutionary runs if pruning is carried out frequently during the generations. We also see that individuals with short genomes of a fixed length - which would theoretically be sufficient to reach high quality solutions - show significantly lower performance than individuals with redundant genomes of a variable length.

In a second set of experiments, the values for structural and functional redundancy are observed, which stay at a constant level of about 20% (70%) for most evolutionary runs. From the analysis of two experiments where all functionally redundant genes were removed after generation 100, it seems that the evolutionary process tries to increase redundancy if additional genetic 'raw material' can have a selectional advantage.

Furthermore, a new measure related to redundancy which was termed *functional proximity* is analyzed. I was able to relate an increase of functional proximity during evolution to more successful evolutionary runs. The last two results indicate that redundancy (and the functional proximity) seem to be the subject of selection.

Of course the absolute values of functional redundancy cannot be directly compared to biological estimations. Nevertheless, I am confident that the analysis of the relation between redundancy and indirect selection

for biological data could reveal interesting insights into biological evolution. Furthermore, for the computer experiments, it would be interesting to study the influence of a trade-off between redundancy and efficiency, if larger genome length are penalized with an energy cost term for reproduction.

This and the previous two chapters contain three analyzes of the performance of the development and the evolution. An application where the individual (or animat) should swim is presented in the next chapter. The morphology is again controlled by the GRN and a control chromosome is added.

Part II

Development of Morphology and Control

9 Simulating the Development of Spiking Neural Networks

Understanding the evolution of biological neural organization using computational models has attracted increasing attention. Two issues are considered essential for this body of research to be biologically plausible. First, the evolution of neural organization should be coupled with that of the body plan [34, 36, 68]. Second, the influence of neural development on the evolution of the nervous system should also be taken into account [108].

The previous chapters dealt with the analysis of GRNs, while the following three chapters concentrate on the function of the developed individuals. In this chapter, the development of the nervous system is simulated and evolved, while Chapter 10 concentrates on the evolution of the morphology of individuals. In contrast to previous chapters, the functionality of the individuals is evolved. The development of both the morphology and the control of individuals is combined in Chapter 11.

A computational model for neural development based on a GRN is suggested in this chapter. Since the *Hydra* is phylogenetically the first animal with a nervous system (see Chapter 2.1.1), it is used as an example organism and its food catching behavior is simulated. Therefore, the neural development is evolved with the GRN described in Chapter 4.2 with additional SUs for neuron formation and cell movement and the food catching ability of the resulting individual is evaluated.

The first task for the gene regulatory model is to achieve a correct developmental order, which takes place in the development of the real *Hydra*. First the stem cells divide, then the interstitial cells migrate, and finally neurons are formed and synapses grow [112]. In the second task, the neurons are distributed over the surface of a cylinder that simulates the body of hydra. A biologically plausible implementation of neural networks, the spiking neural networks (SNNs) are used to simulate the behavior. The neurons are then connected according to a probability inversely proportional to the distance between them. In order to achieve a food catching behavior, the connectivity and weights of the developed neural systems are further adapted. Parts of this work are published in [64].

The food catching task used in this chapter was inspired by the work of Jones et al. [69]. They presented a new 'Hydramat' framework and evolved a spiking neural network to control the movements of a simulated *Hydra* that should catch food. Each individual gained energy when food is caught, it lost energy for each spike in a neuron. Jones et al. used a direct encoding of the positions of the neurons and some connectivity parameters. They demonstrated how efficient information processing and the minimization of energy consumption has emerged in the evolution through the interactions with the environment.

This chapter starts with a short description of neural networks and some models that simulate their behavior. It is followed by a description of the modifications of the GRN model. The experiments and the results are divided into two parts, the first part is about the developmental order and the second illustrates the evolution of the proper food catching task.

9.1 Neural Networks (NNs)

The nervous system controls the behavior of animals and is an extremely complex structure that consists of e.g. connected neurons. They evolved in millions of years starting with few connected simple neurons to high developed structures like the brain.

A lot of literature about nervous systems with different focuses exists. One focus is understanding biological neural networks, and therefore the whole nervous systems. Therefore, a lot of data from biological nervous systems is collected but also models of these systems are presented. Another focus on NNs is to simulate them to solve technical problems [136]. These models are in most cases more abstract and are called artificial neural networks (ANNs). They exist on different levels of abstraction.

In this chapter a biological motivated model of NNs, the spiking neural networks (SNNs) are used to develop individuals that perform food catching behavior. In Chapter 11, central pattern generators (CPGs), which are a more abstract model of NNs, are used to simulate swimming individuals (a more detailed description can be found in Chapter 11).

9.1.1 Spiking Neural Networks (SNNs)

The timing of single spikes (nerve impulses) are used in SNNs to carry information [110] in contrast to more abstract models that e.g. simulate

thresholds only. Networks of spiking neurons are computationally more powerful than these more abstract models [92].

Single-compartment models describe the membrane potential of a neuron by using only one variable $V(t)$, whereas multi-compartment models use a more complex description [32]. For the latter type, the membrane potential $V(x,t)$ can vary over the surface of a neuron.

Single-compartment models are based on the standard equation for capacitors $Q = C_m V$. This yields

$$C_m \frac{dV}{dt} = \frac{dQ}{dt}. \tag{9.1}$$

Substituting the difference in charge dQ by all inputs and outputs results in the elementary equation of single-compartment models:

$$c_m \frac{dV}{dt} = -i_m + \frac{I_e}{A}, \tag{9.2}$$

where I_e is the electrode current, A is the surface of the neuron, i_m is the total membrane current and $c_m = \frac{C_m}{A}$ is the specific membrane current. By convention, the total membrane current is positive, whenever positive ions leave the neuron. The total membrane current can be defined as a sum over all ion channels j:

$$i_m = \sum_j g_j (V - E_j), \tag{9.3}$$

where E_j is the reversal potential and g_j the specific conductance of a channel. Several single compartment models exist, the integrate-and-fire (IAF) model which is used in this chapter is described in the following.

9.1.2 Integrate-and-Fire Neurons

The integrate-and-fire (IAF) model is a simple, but useful model. For some types of neurons it is a good approximation, for others not [32]. In the passive IAF model, the total membrane current i_m of Equation 9.3 is simplified to the single term:

$$i_m = \overline{g}_L (V - E_L), \tag{9.4}$$

where E_L is the resting potential of the model. Substituting i_m in Equation 9.2 results in:

$$c_m \frac{dV}{dt} = -\overline{g}_L (V - E_L) + \frac{I_e}{A}. \tag{9.5}$$

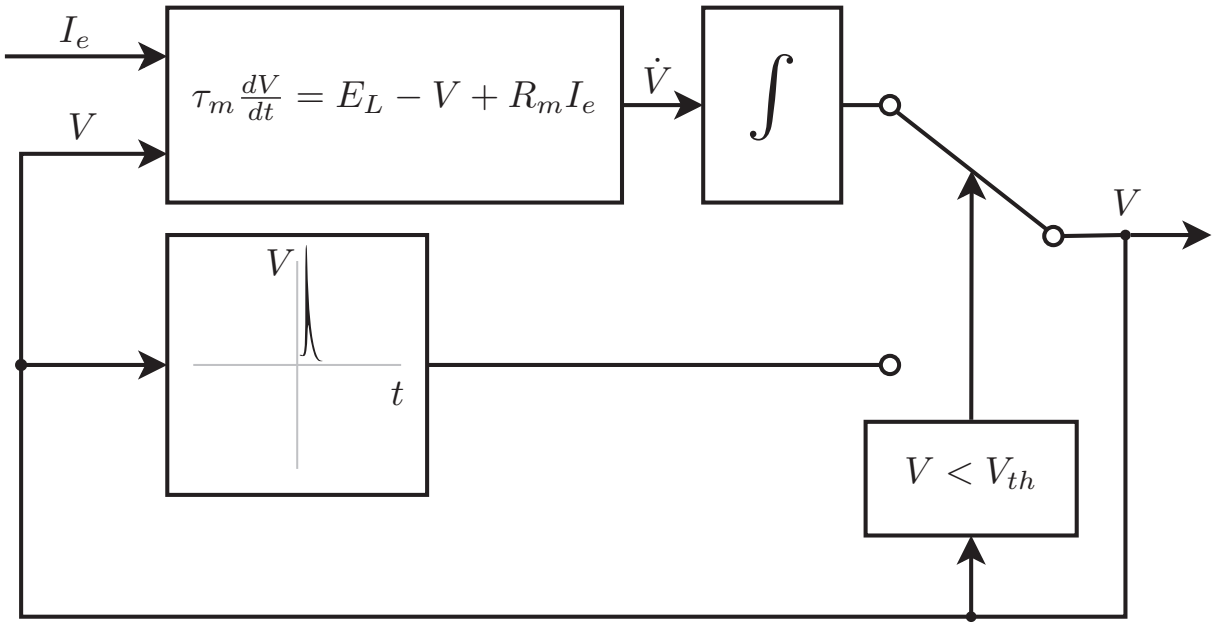


Figure 9.1: Block diagram of a single IAF neuron.

Multiplying Equation 9.5 with $r_m = 1/\overline{g}_L$, substituting $c_m r_m = \tau_m$ and $r_m/A = R_m$ results in the basic equation of passive IAF models:

$$\tau_m \frac{dV}{dt} = E_L - V + R_m I_e, \tag{9.6}$$

where τ_m is the membrane time constant and R_m is the total membrane resistance.

When V reaches the threshold V_{th}, a spike is fired by this neuron and $V = V_{reset}$. In the basic integrate-and-fire model a refractory effect is excluded. The model used in the simulations fixes the potential V of a neuron at its reset potential for the refractory period τ_{ref} immediately after the neuron has produced a spike. After this period, the integration over all inputs is restarted. The block diagram of an IAF neuron is shown in Figure 9.1.

Biologically realistic values for the constants of this model are shown in Table 9.1. The values differ of course for each type of neurons. They were also used for the simulations in this thesis.

In an IAF neural network, the membrane potential of a neuron (V_k) can be calculated by:

$$V_k(t) = \sum_{i=1}^{H} w_{ik} y_i(t), \tag{9.7}$$

Table 9.1: Parameters of IAF neurons.

Variable	Default Value
τ_m	$10[ms]$
V_{th}	$-55[mV]$
V_{reset}	$-70[mV]$
E_L	$-70[mV]$
$R_m = \frac{\tau_m}{C_m}$	$40[M\Omega]$
τ_{ref}	$2[ms]$

where w_{ik} is the weight between neurons k and i, N is the total number of its presynaptic neurons, and $y_i(t)$ is the unweighted contribution of the i-th presynaptic neuron:

$$y_i(t) = \varepsilon(t - t_i^f - d_{ik}), \tag{9.8}$$

where ε is a spike response function modeling the post-synaptic potential, t_i^f is the firing time of neuron i, and d_{ik} is the synaptic delay.

9.1.3 Neural Simulation Technology (NEST)

Several models for neuronal networks exist on different levels of abstractions, a good overview can be found in [104]. The Neural Simulation Technology[1)] (NEST) is a framework designed to simulate large networks of biologically realistic spiking neurons [35, 49]. It is able to simulate neural nets with more than 10^4 neurons. It is a general and efficient method for incorporating precise spike times in globally time-driven simulations and is therefore used for the simulations presented in this chapter.

Many neuron models exist e.g. IAF neurons with many parameters for the different types of neurons that the user can vary. The results of the SNNs that are used in this chapter are simulated with the IAF neurons. Their parameters used here are described in Table 9.1.

9.2 The Model for Neural Development

The developmental model is presented in Chapter 4 with the properties described in Table 9.2. Both ends of the computation area in x direction

[1)] see: www.nest-initiative.org

Table 9.2: Properties of the model used for food catching.

size of computation area	20×100
SUs	SU^{div}, SU^{die}, SU^{TF}, SU^{neuron}, SU^{move}
prediffused TFs	2 with gradients in x and y direction

are connected with each other, so the computation area represents the surface of a cylinder.

Three different cell types are simulated in the model: stem cells, interstitial cells (after division) and neurons (after neuron formation), depending on the activation status of the structural units of the cell. Since one of the tasks is to evolve a developmental order, where first cell division, cell migration and then neuron formation should be executed in all cells, a repairing operator is introduced so that a single gene contains only one type of SU related to cellular behaviors.

The SU for cell division specifies the angle of division, indicating whether the daughter cell is placed above or below the mother cell. For cell migration, two parameters are encoded, one for direction (moving up or down) and the other for moving velocity. The SU for neuron formation encodes the expected lifetime (t_{ttd}) of a neuron and three parameters (c_1,c_2,c_3) determining the probability threshold for synapse growth. Since the neurons in *Hydra* renew each other during the whole lifetime, the lifetime t_{life} of the neurons which is defined by the following Gaussian distribution is limited by:

$$t_{\text{life}} \sim \mathcal{N}(20t_{ttd}, 4). \tag{9.9}$$

The threshold for whether the i-th neuron is to be connected to the j-th neuron is calculated as follows:

$$\varphi_{ij} = \frac{c_1}{1 + e^{c_2 \cdot (d_{ij} - 10c_3)}}, \tag{9.10}$$

where d_{ij} is the distance between the i-th and j-th neuron. The distance is computed in the 2D region (refer to Figure 9.7(b)), which is the distance between the two neurons along the surface of the cylindrical body. Then, a random number p ($p \sim \mathcal{N}(0,1)$) is generated, and if $p < \varphi_{ij}$, a connection between the two neurons will be generated. The connectivity between the interneurons and the sensory neurons, as well as between the interneurons and motor neurons is determined in the same way. Note, however, that there is no direct sensory-sensory, motor-motor, and sensory-motor connection.

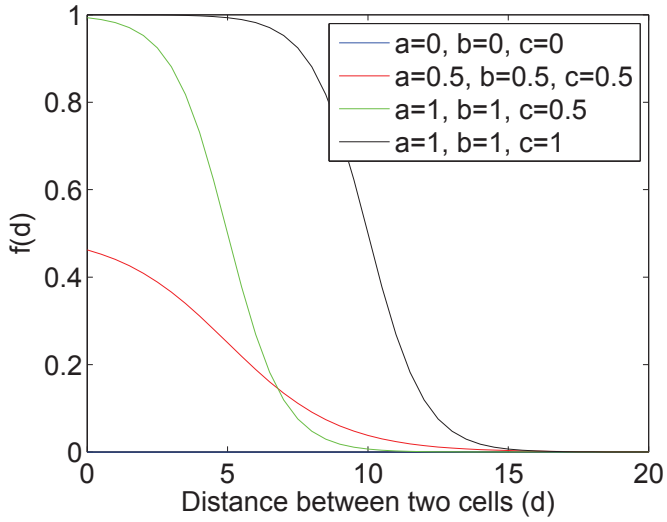


Figure 9.2: Probability function for synapse formation (see Equation 9.10).

9.3 Evolving the Developmental Order

The real *Hydra* has different cell types. Firstly, there are the stem cells, which can divide or differentiate to other cell types, e.g. interstitial cells. The interstitial cells can move or become e.g. a neuron. Therefore a 'correct' developmental order is defined, so that all cells first divide, then move and at last transform to a neuron. Each action can, but not necessarily needs to be active in more than one developmental step.

An evolutionary strategy is applied to evolve the genome to achieve the correct developmental order, i.e., the genes for cell division, cell migration and neuron formation should be activated sequentially during the development. In addition, the gene for division should be deactivated before the gene for migration is activated, as the gene for migration should be deactivated before the gene for neuron formation is activated.

9.3.1 Experiments

First, a simulation of a single cell with only internal TFs is performed. Therefore, division is not executed, so there is only one cell and no cell interactions need to be computed. The assumption made here is that the timing of a cell is an internal process, therefore no external signals are needed (except one constant prediffused TF).

Table 9.3: Properties of the evolution strategy to optimize the developmental order.

μ	1000
λ	3000
σ	$[10^{-6}, 10^{-4}]$
p_{dup}, p_{trans}, p_{del}	0.05, 0.05, 0

The following fitness function is used to penalize 'wrong' developmental orders:

- If two cellular behaviors are active simultaneously, a penalty of 2 is applied.
- If three cellular behaviors are active simultaneously, a penalty of 5 is applied.
- If neuron formation is followed by cell division, 2 is added.
- If neuron formation is followed by cell movement, 1 is added.
- If cell migration is followed by cell division, 1 is added.

The target of the evolution is to minimize the fitness function. The properties of the ES are defined in Table 9.3. The development begins with a single stem cell placed in the center of the simulation area.

9.3.2 Results

The fitness profile is shown in Figure 9.3. We can see that a correct developmental order is achieved after over 300 generations. The large population size and the low selection pressure is needed to find good solutions. Experiments with smaller population sizes failed. The activation levels of the gene for cellular behaviors (refer to Equation 4.10) are shown in Figure 9.6. A value above zero indicates that the corresponding gene is activated and repressed if it drops below or equal to zero.

The timing matrix of an individual with a correct developmental order and cell division turned on again is shown in Figure 9.4. The best individual at the beginning and the end of the development is shown in Figure 9.5. This also shows that the developmental order, which is evolved in a cell with division turned off is still valid when the division is turned on. So

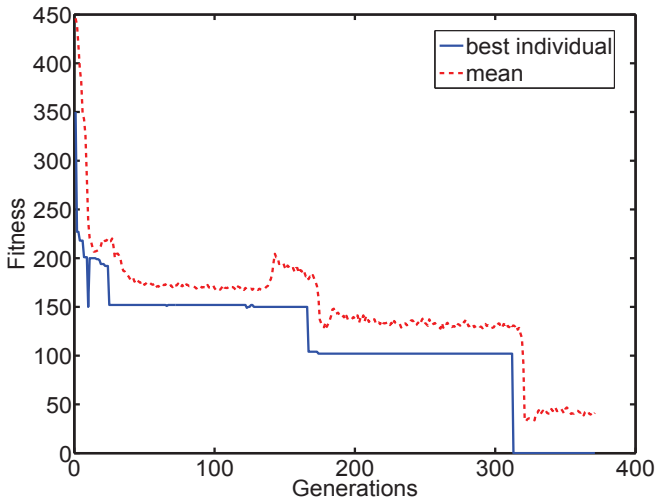


Figure 9.3: Fitness curve of the ES to evolve the developmental order.

the cellular interactions via TFs do not disturb the developmental order.

9.4 Evolution of the Neural Network for Food Catching

The physical body of the hydra-like animat consists of a cylinder with four tentacles equally distributed on its top. Each tentacle is driven by a muscle cell controlled by a neural network composed of motor neurons, interneurons and sensory neurons, all of which are modelled with integrate and fire (IAF) neurons, refer to Figure 9.7. To simplify the experimental setup, four sensory neurons and four motor neurons are put on the top of the body, while the number and position of the interneurons are determined by the developmental process. During the behavior adaptation, four pieces of food drop sequentially around the body and the closest sensory neuron will generate a number of spikes as the input to the neural network. The positions of the four pieces are fixed, so that one piece drops at each 'side' of the cylinder.

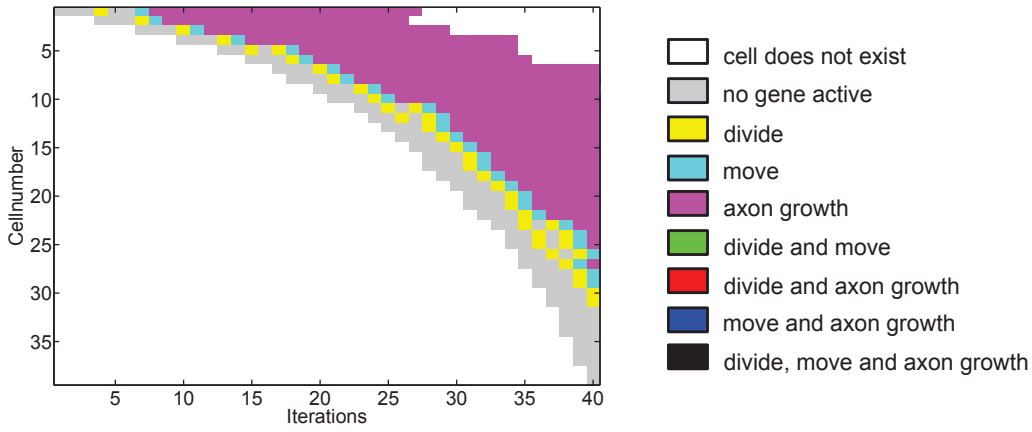


Figure 9.4: Activations of the genes of all cells for all developmental steps with a correct developmental order.

Table 9.4: Properties of the evolution strategy used for food catching.

μ	30
λ	200
σ	$[10^{-6}, 0.1]$
p_{dup}, p_{trans}, p_{del}	0.05, 0.05, 0

9.4.1 Experiments

Twelve stem cells with the genome that performs the cellular behaviors in the correct order are distributed over the simulation area so that a neural network is developed. For the neural network to perform the food catching behavior, an ES to adapt its connectivity and weights is employed. It is assumed that only one tentacle is needed to catch a piece of food. Thus, the target of the behavior adaptation is that the motor neuron closest to the dropping food should fire as strong as possible to maximize the possibility to catch the food, while the activity of other motor neurons should be minimal to reduce energy consumption.

To achieve the above-mentioned target, the following fitness function is defined for the i-th motor neuron (output neuron):

$$F_i = -f_i + \sum_{j!=i} ns_j, \text{for } i,j = 1,...,4, \tag{9.11}$$

where ns_j is the number of spikes generated by other motor neurons in the simulated time period, penalizing unnecessary energy loss, f_i reflects the performance of the i-th neuron measured by the time (t_i^c) for the

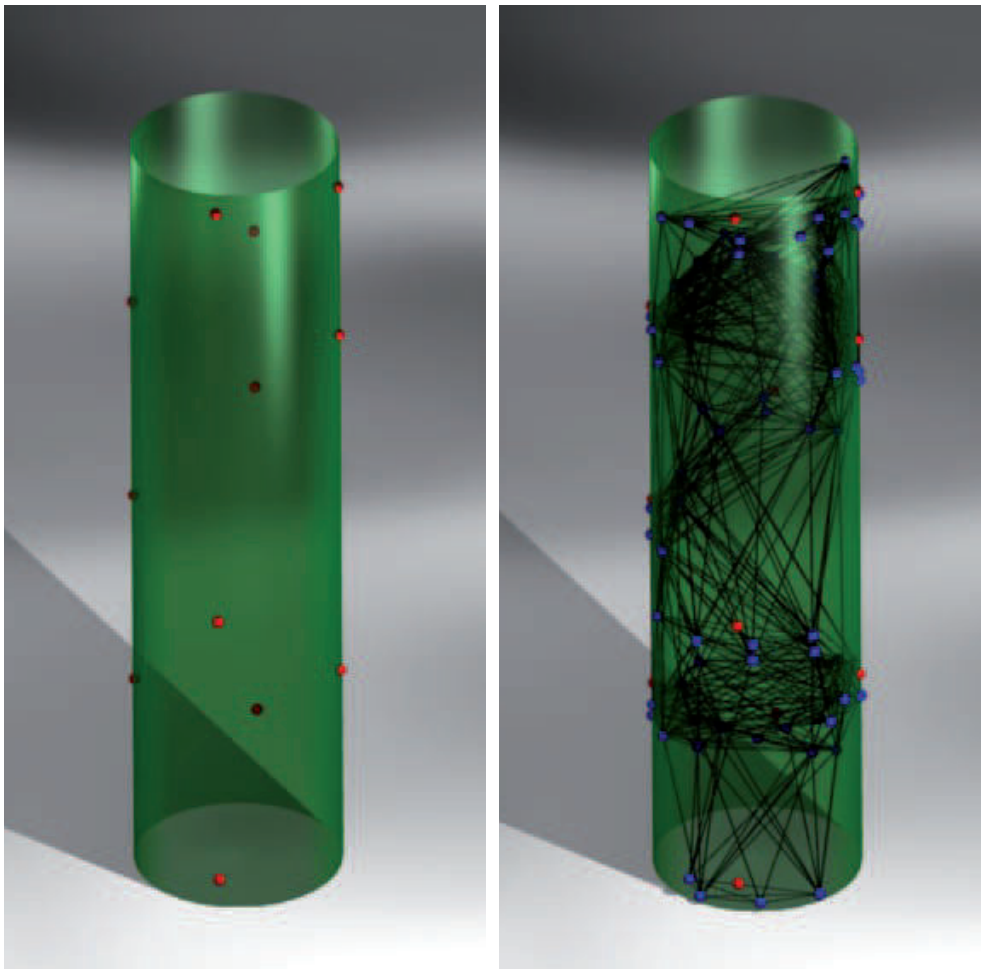


Figure 9.5: The stem cells at the beginning of the development (left) are distributed over the hull of hydra. The resulting individual with all synapses is shown on the right. Stem cells are red and neurons are blue.

corresponding tentacle to stand vertically, i.e., when the angle $\lambda = 90°$, shown in Figure 9.7, or by the maximum λ during the simulation period, if λ never reaches 90° during that time. The angle of the i-th tentacle is calculated as

$$\lambda_i(t) = \lambda_i(t - 0.1) - g \cdot 0.1 + s_i(t),\ \lambda_i \in [0, 90]. \tag{9.12}$$

The second term on the right side of Equation 9.12 simulates the passive dropping-down of tentacle due to gravity (g), $t \in [0, 30]$ is the simulation time in milliseconds in behavior adaptation, $s_i(t)$ equals 0 or 1 depending on whether there is a spike at time t. The final fitness is summed over four runs with different food dropping conditions.

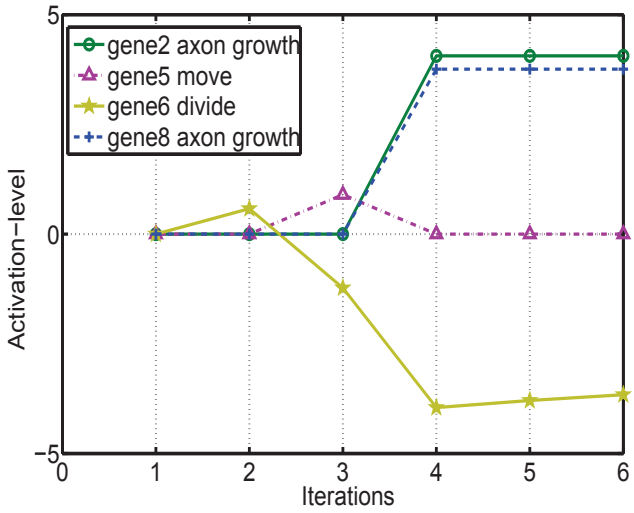


Figure 9.6: Activation levels of the genes of one cell with a correct developmental order.

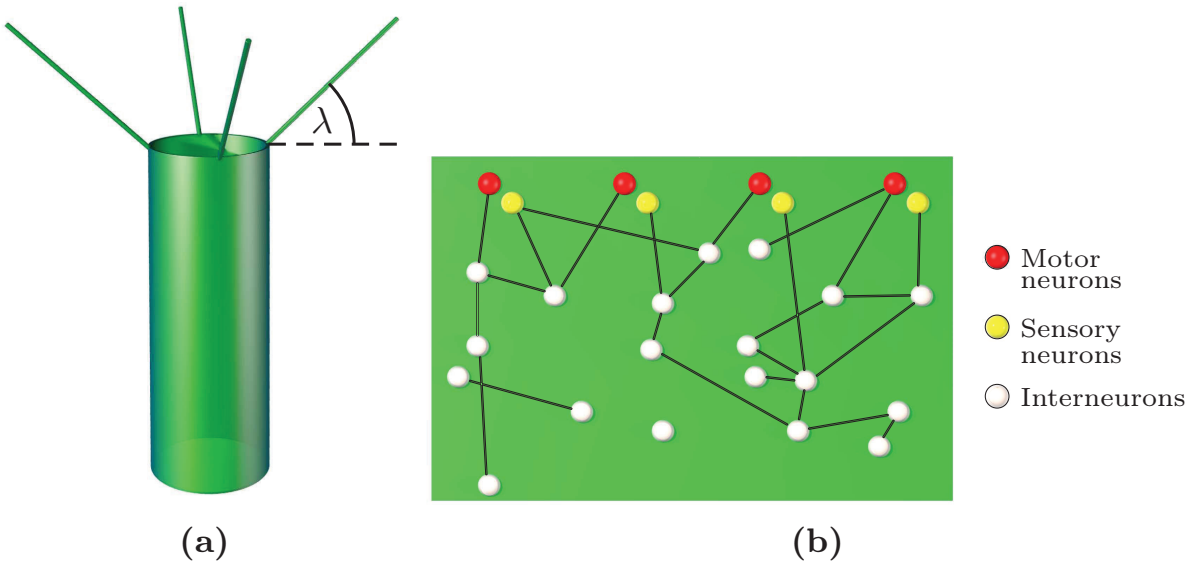
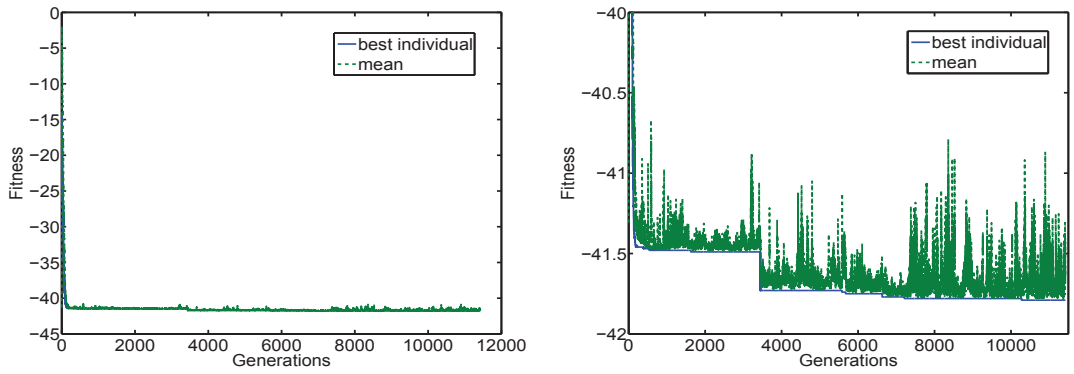


Figure 9.7: The hydra-like animat. (a) The body plan with four tentacles, and (b) the nervous system shown on the 2-D area representing the unfolded body surface.



(a) Fitness profile **(b)** Fitness zoomed in

Figure 9.8: Fitness curve of the evolution of the food catching task.

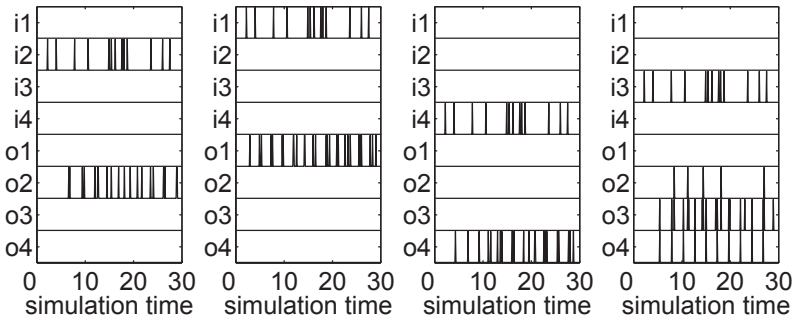


Figure 9.9: Spikes at the input (i1-i4) and output neurons (o1-o4) for four different inputs.

The maximal weight is set to 3000, which can be excitatory or inhibitory, and the synaptic delays are fixed to one. All parameters are scaled between zero and one.

9.4.2 Results

The fitness curves for behavior optimization are shown in Figure 9.8. The spikes of the input and output neurons of the best adapted individual are shown in Figure 9.9. From this, we can see that in three of the four cases, the neural network performs optimally. In one case (last panel in Figure 9.9), outputs 2 (o2) and 4 (o4) also fire, although only output 3 (o3) is assumed to spike. Nevertheless, o3 does fire the strongest.

9.5 Discussion

This chapter suggests a neural developmental model based on a gene regulatory network. The GRN has evolved successfully to achieve the correct developmental order, i.e., cell division, cell migration and neuron formation in sequence in a simulated evolution that undergoes gene transposition, duplication and mutation. After the gene regulated neural development is complete, the connectivity and weights of the neural network are further adapted using an ES to successfully perform a food catching behavior in a hydra-like animat.

The current developmental model could be improved in various respects. The cell migration behavior resulted from the current model is quite deterministic, as there is little local interaction between the cells. This is also caused by the first run where the developmental order is evolved in a single cell and the individual for the food catching task is based on the results of the first evolution.

Contrary to the work of Jones et al., a developmental model is used to grow the positions of the neurons to perform a food catching task. This encoding is biologically more plausible and the number of neurons is not fixed. The energy of the motor neurons is taken into account here, but the energy loss by spikes of the other neurons are not taken into account.

The next two chapters discuss the evolution of swimming individuals, where the morphology and the control are both subject to evolution. Since I want to concentrate on the coevolution of morphology and control, a less complex model of NNs is used in the following.

10 Simulating the Development of Swimming Animats

The individuals evolved in this chapter should swim in a simulation environment and therefore their fitness depends on their morphology and control concurrently. In the first part of this thesis a morphology was evolved, whereas in the previous chapter the functionality of a neural network was optimized. Here, the morphology is controlled by the same GRN as in the first part, for the control of the movements a more abstract encoding is used. Parts of this chapter are based on [114].

This is a first step towards developing both, morphology and control concurrently and in one chromosome. As described in the Introduction, this can improve the optimizations because of the effects of coevolution. In the existing models mentioned in Chapter 3.3, the developmental process of the neural system or the morphology is simulated. If both is included in the models, they are optimized separately. In the next chapter, the control of the movements is realized by a simple neural network which is also developed. There, both the morphology and control of the individual are defined in the GRN which provides many possibilities for the optimization method.

This chapter starts with a description of the model, i.e. with the encoding of the movements and the environment for swimming. This is followed by a specification of the experiments and a discussion of the results.

10.1 The Model for Swimming Animats

The evolutionary strategy and the model of the development of the morphology have already been described in Chapter 4, the parameter setting is given in Table 10.1. The next section contains a specification of the motor control and is followed by the description of the simulation environment for the swimming animats.

Table 10.1: Properties of the model used for evolving swimming animats.

size of computation area	80×80
SUs	SU^{div}, SU^{die}, SU^{TF}
prediffused TFs	2 with gradients in x and y direction
μ	30
λ	200
σ_m	$[10^{-6}, 10^{-4}]$
σ_c	$[10^{-6}, \infty]$
p_{dup}, p_{trans}, p_{del}	see Table 10.3

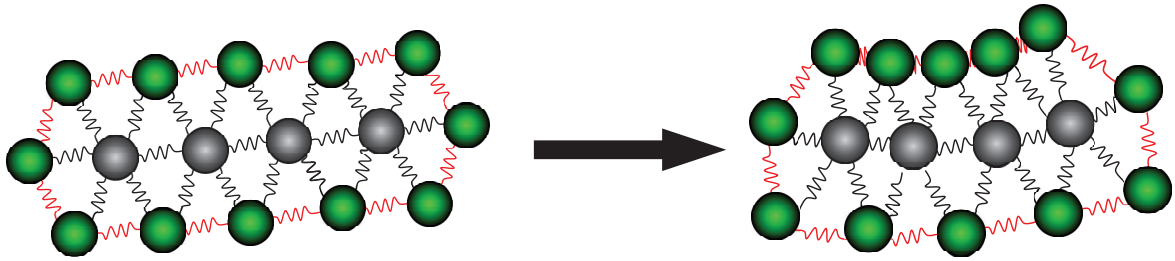


Figure 10.1: Illustration of a body plan consisting of cells connected by springs. The springs at the outside of the body are able to change their natural length.

10.1.1 Chromosome for Motor Control

To embed a motor controller into the developed morphology, cells must be connected to a whole body plan. Cells are connected with a damped spring if the distance between them is smaller than 2.5. If a cell has less than two connections, it is connected to its nearest neighbor to ensure morphological stability as shown in Figure 10.1. The mechanical setups of the cells and springs are listed in Table 10.2.

The movement is defined by a change in the natural length of the springs connecting the cells on the outside, as depicted in green in Figure 10.1. To ease the movement, the cell radius is set to 0.5 so that there is sufficient space between the cells for them to move. The natural length of the springs switches between l_n and l_s within one period T, which is subject to evolution. The morphology is split into 24 predefined segments and all springs in the same segment have the same phase shift ($\rho \in [0,T]$).

Table 10.2: Constants for the mechanical simulation environment.

mass of cells m	0.5
radius of cells r	0.5
damping constant d	1
spring strength c	5
normal natural length of springs l_n	2
short natural length of springs l_s	1.2
minimal periodic time T_{min}	10
maximal periodic time T_{max}	400
simulation length t_{sim}	300.0

10.1.2 Physics Simulation

The physics simulation engine used to simulate the behavior of the animats is BREVE [80]. A simple model for simulating the effects of water forces is added, which has also been adopted in [117]. A detailed description of the water forces and the simulation environment is given in Appendix C. In this model, the water forces for different elements i are computed as follows:

$$\boldsymbol{F}^i = \boldsymbol{F}^i_T + \boldsymbol{F}^i_N, \tag{10.1}$$

$$\boldsymbol{F}^i_T = -\lambda_T \cdot sgn(\boldsymbol{v}^i_T) \cdot (\boldsymbol{v}^i_T)^2, \tag{10.2}$$

$$\boldsymbol{F}^i_N = -\lambda_N \cdot sgn(\boldsymbol{v}^i_N) \cdot (\boldsymbol{v}^i_N)^2, \tag{10.3}$$

where λ_T and λ_N are the drag coefficients for each direction. λ depends on the effective area, a shape coefficient of the element and the fluid density. $\boldsymbol{v}^i_T$ and $\boldsymbol{v}^i_N$ are the velocities of element i in normal and tangential direction. $\lambda_T = 0.001$ and $\lambda_N = 2.5$ are used in this work. The water forces are computed for cells in the lateral of the body plan, represented by green circles in Figure 10.1. The normal and tangential vectors of the body parts (i-th sphere) can be calculated by:

$$\boldsymbol{t}^i = \frac{\boldsymbol{p}^{i-1} - \boldsymbol{p}^{i+1}}{|\boldsymbol{p}^{i-1} - \boldsymbol{p}^{i+1}|}, \tag{10.4}$$

$$\boldsymbol{n}^i = \begin{pmatrix} 0 & -1 \\ 1 & 0 \end{pmatrix} \cdot \boldsymbol{t}^i, \tag{10.5}$$

Table 10.3: Three setups of the ES so evolve swimming animats.

	initial # RUs	initial # SUs	p_{transp}	p_{dup}	p_{del}
setup 1	66	54	0.05	0.05	0
setup 2	30	30	0.05	0.05	0
setup 3	30	30	0.05	0.02	0.03

where $\boldsymbol{p}^i$ is the position vector of the i-th cell and $\boldsymbol{p}^{i-1}$ and $\boldsymbol{p}^{i+1}$ are the positions of the neighboring cells on the outside of the morphology.

$$\boldsymbol{v}_N^i = \boldsymbol{n}^i \cdot \boldsymbol{v}^i, \tag{10.6}$$

$$\boldsymbol{v}_T^i = \boldsymbol{t}^i \cdot \boldsymbol{v}^i, \tag{10.7}$$

where $\boldsymbol{v}^i$ is the velocity of the i-th cell.

10.2 Experiments

Different strategy parameters (σ_m and σ_c) are used for the two chromosomes for morphological development and for behavior control, respectively. For the chromosome that controls morphological development gene duplication and transposition are used in addition to mutation as described in Chapter 4. In one setup, gene deletion is also applied, refer to Table 10.3.

The fitness function is twofold, it contains a part representing the distance the individual swims and a part that rewards elongated shapes. The fitness for shape is similar to those in Chapter 5. The following fitness function is minimized:

$$f = f_{swim} + f_{shape}, \tag{10.8}$$

where f_{swim} defines the distances between the centers of masses of the body plan at the beginning and the end of the simulation:

$$f_{swim} = -\left| \left(\sum_{i=0}^{n} \boldsymbol{x}^i(t=0) \right) - \left(\sum_{i=0}^{n} \boldsymbol{x}^i(t_{end}) \right) \right|. \tag{10.9}$$

f_{shape} awards elongated shapes:

$$\begin{aligned} f_{shape} = \max\left\{ \min_i \left\{ \boldsymbol{x}^i(0) \right\}, -30 \right\} - \min\left\{ \max_i \left\{ \boldsymbol{x}^i(0) \right\}, 30 \right\} \ldots \\ \ldots - \min\left\{ \min_i \left\{ \boldsymbol{x}^i(1) \right\}, -5 \right\} + \max\left\{ \max_i \left\{ \boldsymbol{x}^i(1) \right\}, 5 \right\}, \end{aligned} \tag{10.10}$$

where the best reachable value for f_{shape} is -50.

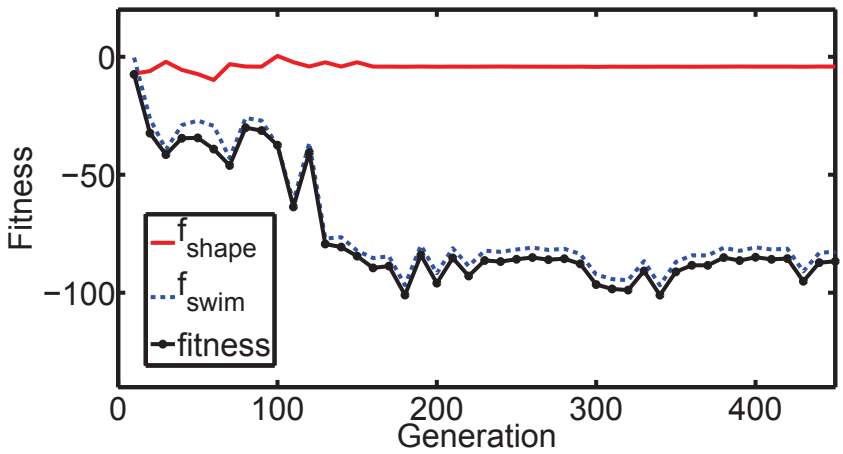
To limit the computational cost, a maximum of 501 cells is allowed. If this number is exceeded in less than 20 iterations, the developmental process is stopped after this developmental step. To obtain a meaningful morphology, the number of cells should be larger than 10. When the number of cells is larger than 500 or smaller than 10, a strong penalty is applied to the fitness.

The developmental process is computed for $t_{dev} = 20$ iterations. Three slightly different setups, which are listed in Table 10.3, are used for the three evolutionary runs.

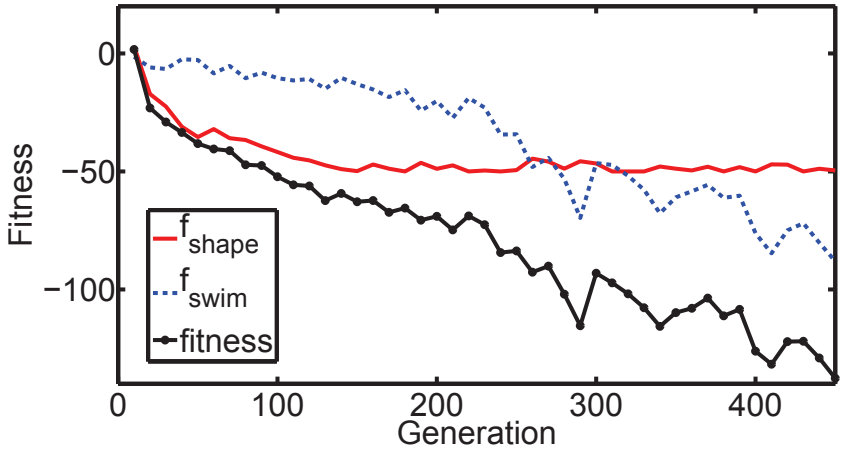
10.3 Results

In the following, the fitness curves of the setups, the morphology and the control strategy of three animats, are analyzed. A good animat from each setup is picked out, they were taken from generation 1518, 541 and 197, respectively. The fitness for swimming behavior f_{swim} and that for morphology f_{shape} is plotted separately every tenth generation in Figure 10.2. The f_{shape} of the three animats are -4.21, -50.0, and -9.42, while the f_{swim} are -97.38, -107.50, and -84.87, respectively. Figure 10.2, shows that the fitnesses for the shape in both setup 1 and 3 have converged to a local minimum around generation 150. As a result, the morphology from these two setups is much smaller than that obtained from setup 2, refer to Figure 10.3. Another observation is that the fitness for swimming in setup 2 improves steadily in 450 generations and is better than that from setups 1 and 3. Actually, the best animat evolved in setup 2 reached the border of the simulation area within the predefined simulation time.

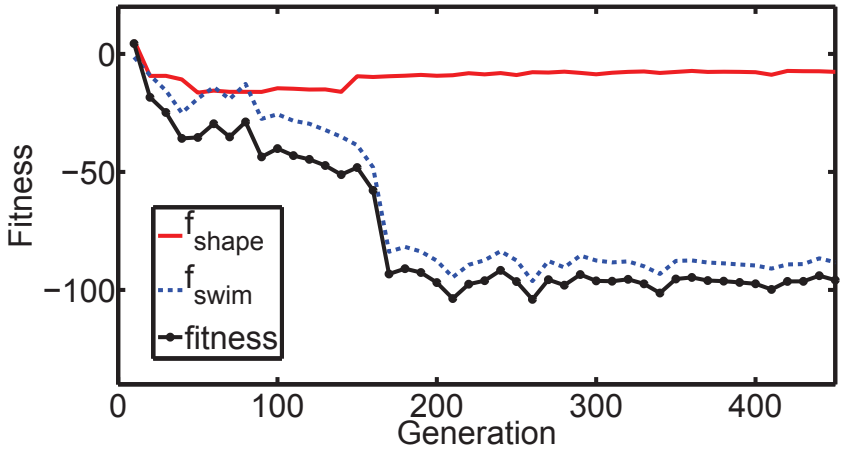
More interestingly, the animats have also evolved different strategies for swimming. In setups 1 and 3, where the evolved morphology is very short, a control period of $T = 10$ is evolved. By contrast, a period $T = 16.3$ has been evolved for the animat in setup 2. In other words, the frequency of the rhythmic movement of the shorter animats is much faster than that of the large one, which makes sense for improving the swimming efficiency. In addition, different phase coordination strategies have also been evolved for the three animats, as shown in Figure 10.4. From the phase shift patterns, we can see that animats from setups 1 and 2 produce undulatory movements, while the animat from setup 3 generates peristaltic movements, similar to a caterpillar. A few snapshots of the resulting movement patterns of the three animats are presented



(a) Setup1



(b) Setup2



(c) Setup3

Figure 10.2: Fitness profiles from the three setups, which are plotted every tenth generation.

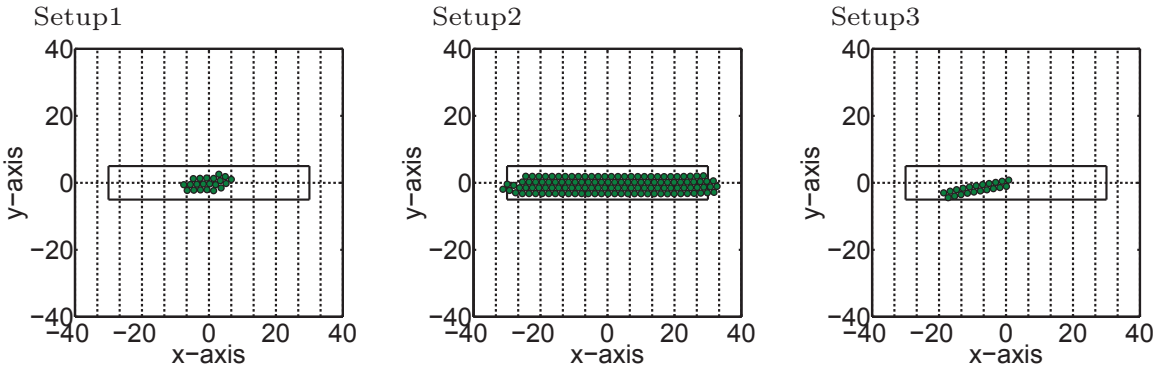


Figure 10.3: Morphologies of the three individuals evolved for swimming.

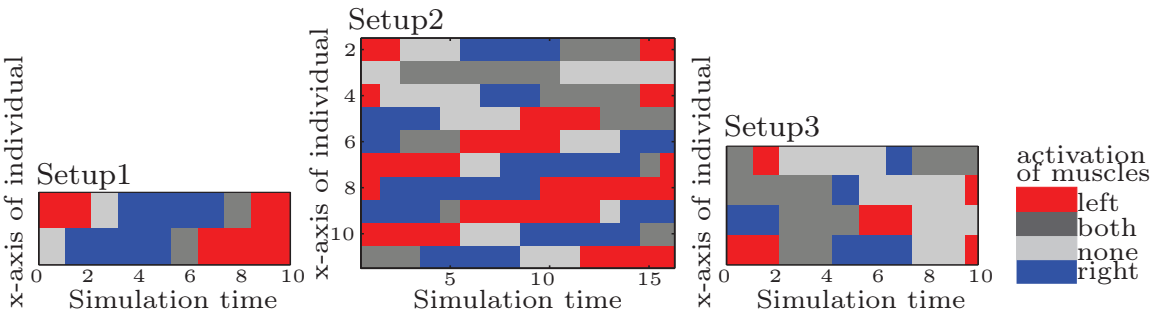


Figure 10.4: Evolved phase coordination strategies of the three individuals.

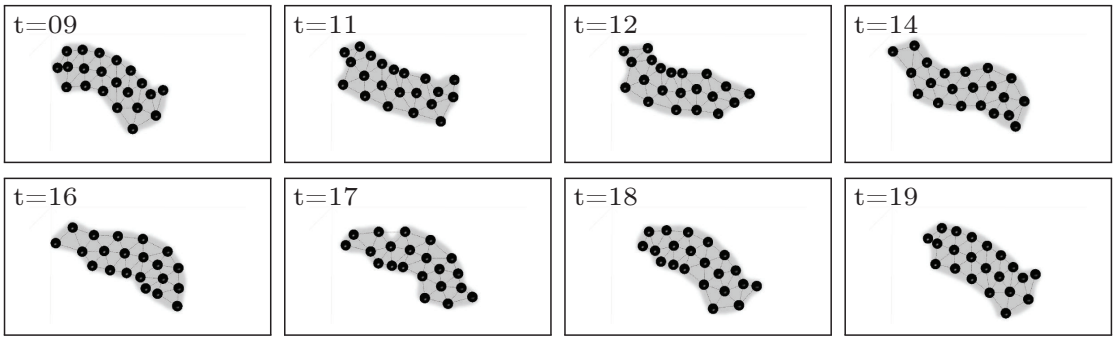


Figure 10.5: Snapshots of the movement of the analyzed individual from setup 1.

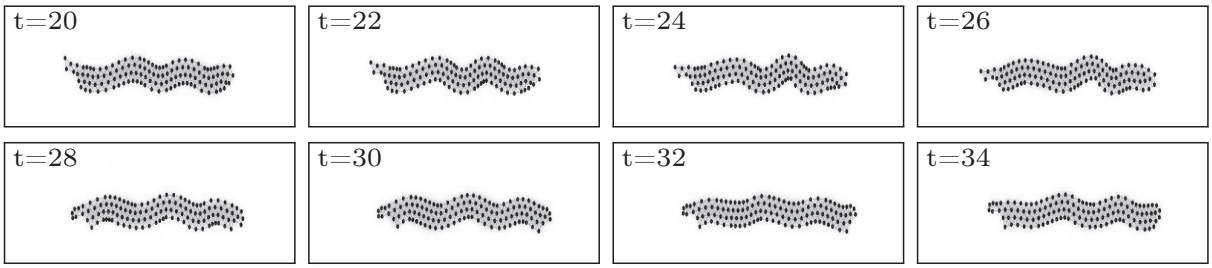


Figure 10.6: Snapshots of the movement of the analyzed individual from setup 2.

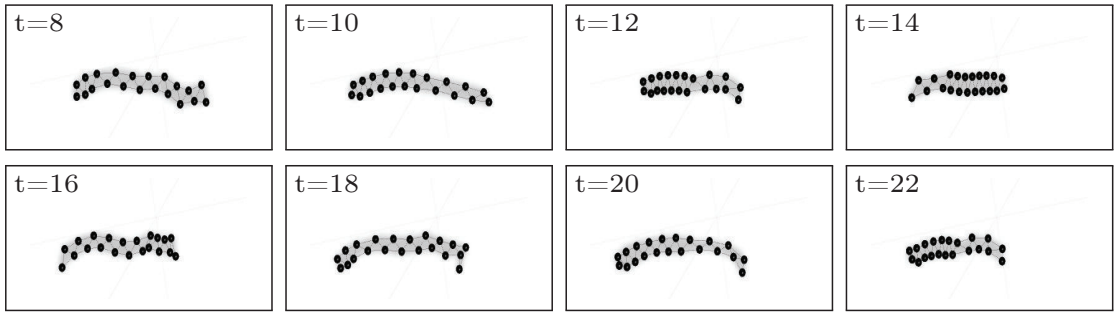


Figure 10.7: Snapshots of the movement of the analyzed individual from setup 3.

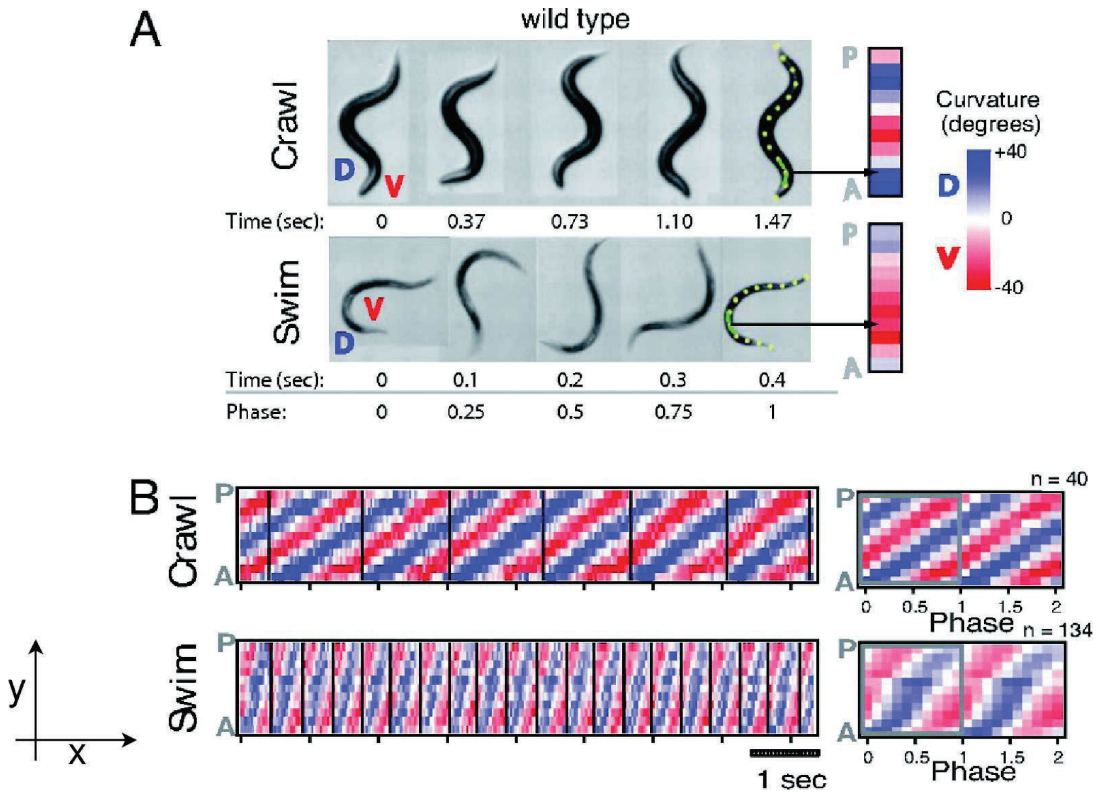
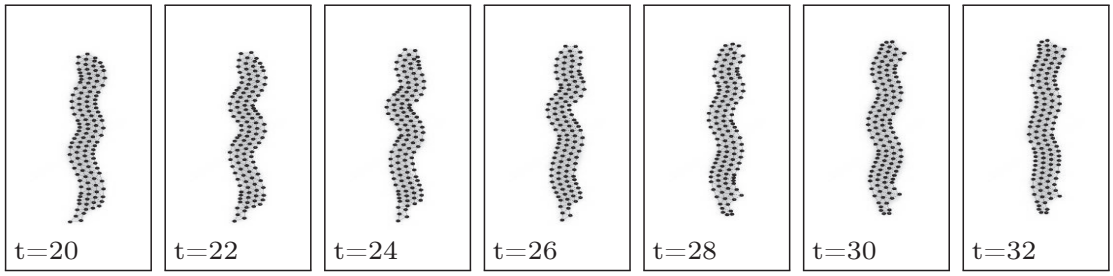


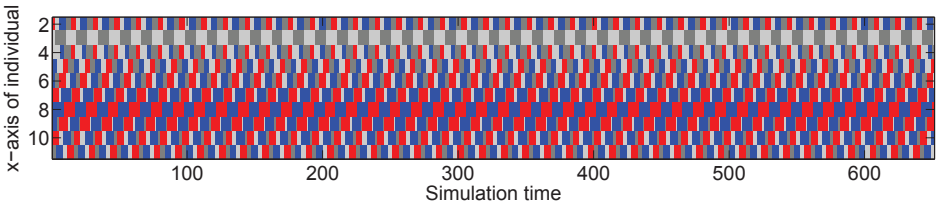
Figure 10.8: Subfigure (A) shows snapshots of a swimming and a crawling *C. elegans*. The patterns of their movements are illustrated in (B). The y-axis represents the individual from head to tail, the x-axis the time, and the colors symbolize the curvature of the body part. Reprinted from [107]. Copyright (2008) National Academy of Sciences, U.S.A

in Figure 10.5, Figure 10.6, and Figure 10.7 (videos are available at `www.rtr.tu-darmstadt.de/coevolution`). Although the goal of the evolution strategy was not the pattern of a specific animal, a small comparison of the individual evolved in setup 2 with the movements of *C. elegans* follows.

Pierce-Shimomura et al. analyze the swimming and crawling behavior of *C. elegans*, see Figure 10.8 [107]. The same arrangement for setup 2 is presented in Figure 10.9 and Figure 10.10 shows the phase coordination behavior of the animat evolved in setup 2 and the swimming pattern of *C. elegans*. There are minor differences, e.g. there is one period over the body length in *C. elegans* and more than one period in the simulated individual. Additionally, at the one end of the simulated individual (low x values) the sine curve is less accurate. The swimming patterns are altogether comparable, although only the swimming of an elongated shape was subject to the artificial evolution.

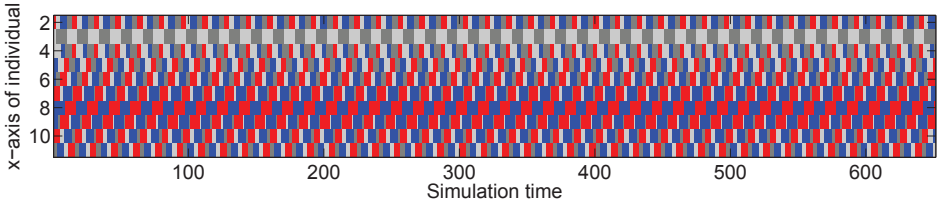


(a)

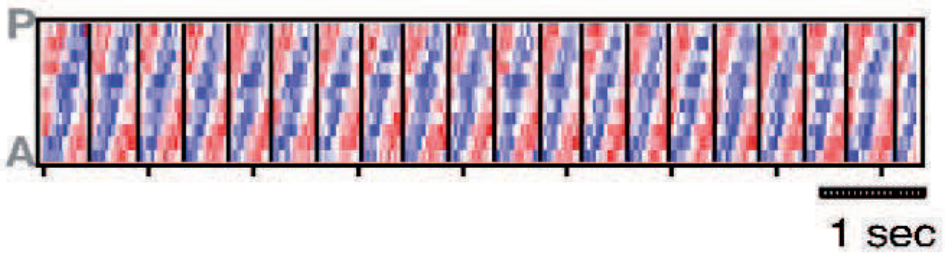


(b)

Figure 10.9: To compare the movements evolved in setup 2 with those of *C. elegans* this figure has the same arrangement as Figure 10.8. Subfigure (a) shows the swimming individual and the patterns of its movements are illustrated in (b).



(a) Scaled pattern evolved in setup 2.



(b) Pattern of a swimming *C. elegans* reprinted from [107]. Copyright (2008) National Academy of Sciences, U.S.A.

Figure 10.10: Comparison of the swimming pattern evolved in setup 2 and the pattern of a swimming *C. elegans*.

10.4 Discussion

A model for co-evolving morphological development and motor control for swimming animats has been presented in this chapter. The morphological development is based on a cellular growth model regulated by a GRN, whilst the motor control is represented by period and phase shifts of the springs. Compared to the direct graph model used by Sims [120] and the L-system of Hornby and Pollack [60], the model for morphological development presented here is biologically more plausible. This can result in shapes and swimming patterns that are closer to real animals, as shown in the comparison with *C. elegans*.

From three slightly different setups of the evolutionary algorithm, three swimming patterns have emerged, which have adapted to the different morphologies. A gene-regulated multi-cellular model for morphological development of animats that can perform a functional behavior and to disclose a coupling between the motor control strategy and the body plan is presented.

Here, two separated chromosomes for morphological development and control are used. In the next chapter a GRN-based model for neural development is combined with the one for morphological development, so that both, the neural system and body morphology are subject to a single developmental process. The GRN-based model for neural development described in the previous chapter is simplified to provide the opportunity to evolve the morphology and the control in one chromosome.

11 Coevolution of Morphology and Control for a Swimming Animat

The development of a nervous system has been presented in Chapter 9 and in the previous chapter the development of the morphology has been extended to provide a certain functionality, in this case swimming, using a simple control structure. In this chapter, the development of control and functionality is combined, however a more abstract form of nervous systems is used. Using central pattern generators facilitates the evolution of an oscillating movement, which is helpful for the swimming task. Evolving the developmental order for neuron formation, which is a constraint in Chapter 9, is omitted here to make the task less complex.

Designing shape and control concurrently should improve the design process. An optimal shape can reduce the complexity of the control part, e.g. passive dynamic walker can walk down a slope without control only because of its mechanical configuration, which means they are capable of stable, human-looking walking [94]. Depending on the task, an optimal trade-off can be found. In standard design processes, first the morphology of e.g. a robot is fixed and then the control is defined. Optimizing the morphology and the control at once provides the opportunity to find the optimal trade-off between both and to examine interesting evolutionary interaction patterns on the genetic level. The system presented here can be used to design the shape and the control of robots etc. at once and therefore improve the design process. To use the same genome for shape and control could also improve this process further, because as in biology effects of coevolution can occur. This is contrary to most existing models that use a separated genome for the morphology and the control as presented in Chapter 3.3. Parts of this work are published in [113].

The chapter starts with an introduction to central pattern generators (CPGs) in general and a more detailed description of the CPG model used in this chapter. The next section is used to define modifications to the

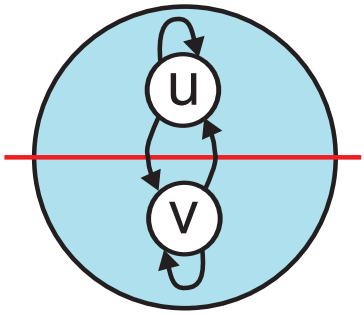


Figure 11.1: The model of a central pattern generator contains two neurons that interact with each other.

GRN model. The experiments are defined in Section 11.3 and the results are subsequently presented in Section 11.4. The chapter is completed with a discussion.

11.1 Central Pattern Generators (CPGs)

Mirollo and Strogatz analyze the synchronization of pulse-coupled biological oscillators. They compare a model of integrate-and-fire oscillators with several real oscillators e.g. flashing fireflies, crickets that chirp in unison and some more [101]. Many animals use coupled rhythmic muscle activations for movements. This movement is not controlled by the brain, but by coupled oscillators, the central pattern generators (CPG). They can be found for example in dogfish or lamprey. It can be shown, that the pattern occurs also after the spinal cord has been separated from the brain [102].

Several models of CPGs exist, e.g. [13, 25, 62, 102, 135], in general the CPG consists of two neurons which interact with each other, see Figure 11.1. The difficulty with most models is the stability of the output of many CPGs depending on their connections. The output of the CPGs should ideally be sinusoidal with phase shifts between the output signals of the different CPGs depending on their synapse connections and weights.

Chung and Slotine use coupled Hopf-Kuramoto oscillators and show their ability to synchronize almost globally [25]. This model is used for the experiments presented in the following because of its good ability to

Table 11.1: Properties of the CPG Model.

k	0.01
ω	0.3
ρ	1
λ	1
σ	1

Table 11.2: Additional SU for neuron formation. All other SUs are described in Table 4.2.

	Neuron Formation
s_1	[0.6, 0.8]
s_2	-
s_3	-
s_4	TF affinity aff_{TF} for CPG orientation
s_5	-
s_6	probability values
s_7	for axon growth
s_8	c_1, c_2, c_3
s_9	-

synchronize. Therefore, $\boldsymbol{x}_i(t) = (u_i(t),v_i(t))^T$ and the following equations are used:

$$\dot{\boldsymbol{x}}_i = \boldsymbol{f}(\boldsymbol{x}_i;\rho_i) - k\sum_{j\in\mathcal{N}_i}^{m_i}\left(\boldsymbol{x}_i - \frac{\rho_i}{\rho_j}\boldsymbol{R}\left(\phi_{ij}\right)\boldsymbol{x}_j\right) \tag{11.1}$$

$$\boldsymbol{f}(\boldsymbol{x};\rho) = \begin{pmatrix} -\lambda/\rho^2\left(u^2+v^2-\rho^2\sigma\right)u - \omega(t)v \\ \omega(t)u - \lambda/\rho^2\left(u^2+v^2-\rho^2\sigma\right)v \end{pmatrix}. \tag{11.2}$$

The properties of the model for the simulations in this thesis are described in Table 11.1.

11.2 Gene Regulatory Model

The gene regulatory network model described in Chapter 4 is extended with an additional SU for neuron formation, see Table 11.2. A cell with an active gene for neuron formation becomes a CPG for the rest of its

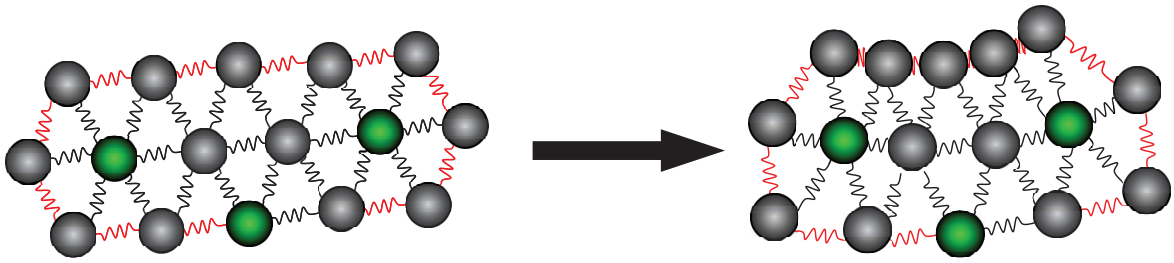


Figure 11.2: Illustration of a body plan consisting of cells connected by springs. The CPGs are depicted in green. The springs on the outside of the body (red) are able to change their natural length, except the springs associated to a CPG.

lifetime. All cells on the outside of the individual that are not CPGs at the end of the development are termed muscle cells. The axon formation between the CPGs is defined as in Chapter 9 and depends on the distance between the CPGs (see Equation 9.10 and Figure 9.2).

The muscle cells contract with the output of one of the neurons of the closest CPG. When the distance to the closest CPG is higher than 8, the muscle cell is passive. A contraction of a muscle cell means a change in the rest length of the associated spring at the outside of the individual (counter-clockwise).

Since each CPG contains two neurons (u and v), an orientation of the CPG is introduced to define to which neuron a cell is connected. The orientation of the CPG itself is defined by the gradient of a TF, which TF is used is defined in the SU for neuron formation. Parameter s_4 in the SU defines an affinity value, the TF with the closest affinity to the affinity encoded in s_4 is used for the orientation of the CPG. Cells which connect to the CPG between $0 - 180°$ are connected to the neuron u and cells connected with an angle of $180 - 360°$ are connected to the neuron v of the CPG.

11.3 Experiments

The goal of the experiments is to evolve individuals that swim the furthest in a desired time. The fitness function for swimming is defined as in Chapter 10:

$$f_{swim} = -\left|\left(\sum_{i=0}^{n} \boldsymbol{x}^i(t=0)\right) - \left(\sum_{i=0}^{n} \boldsymbol{x}^i(t_{end})\right)\right|, \tag{11.3}$$

Table 11.3: Properties of the model used for the experiment.

size of computation area	100×80
SUs	SU^{div}, SU^{die}, SU^{TF}, SU^{neuron}
prediffused TFs	2 with gradients in x and y direction
μ	45
λ	300
elitists	3
initial number RUs and SUs	50, 50
σ	10^{-4}
p_{dup}, p_{trans}, p_{del}	0.05, 0.03, 0.02

so the center of mass of the individual at the beginning and the end of the swimming period are computed and the distance is calculated.

As described in Chapter 5, the size of the individuals is limited, so the number of cells (n_c) is constrained between 10 and 500. A penalty of $600 - n_c$ will be applied if $n_c < 10$ and a penalty of n_c if $n_c > 500$. If the cells in the developed morphology are not fully connected, a poor fitness of 100 will be assigned (contrary to the value of 50 in the first part of this thesis).

When the individual consists only of neurons or has no neurons, there will be no movement and the fitness for swimming is therefore set to zero ($fit_{swim} = 0$). If the CPGs are not connected, which means there is no path to another CPG via synapses, the CPGs cannot synchronize and their phase shift is random and therefore depends on the initial values of the differential equation. To avoid that not connected CPGs get established during the evolution, but still not to penalize it too strong, the fitness for swimming is then halved.

The setup is defined in Table 11.3, four different runs with different random seeds are performed. The constants for the mechanical simulation environment are the same as in the previous chapter, see Table 10.2, except the simulation length which is changed to $t_{sim} = 500$.

11.4 Results

The fitness curves of the four different runs are shown in Figure 11.3. The resulting individuals all swim between 53 and 82 length units (53.9, 82.3, 53.5, 62.2). Since the simulation time is longer and the frequency of the

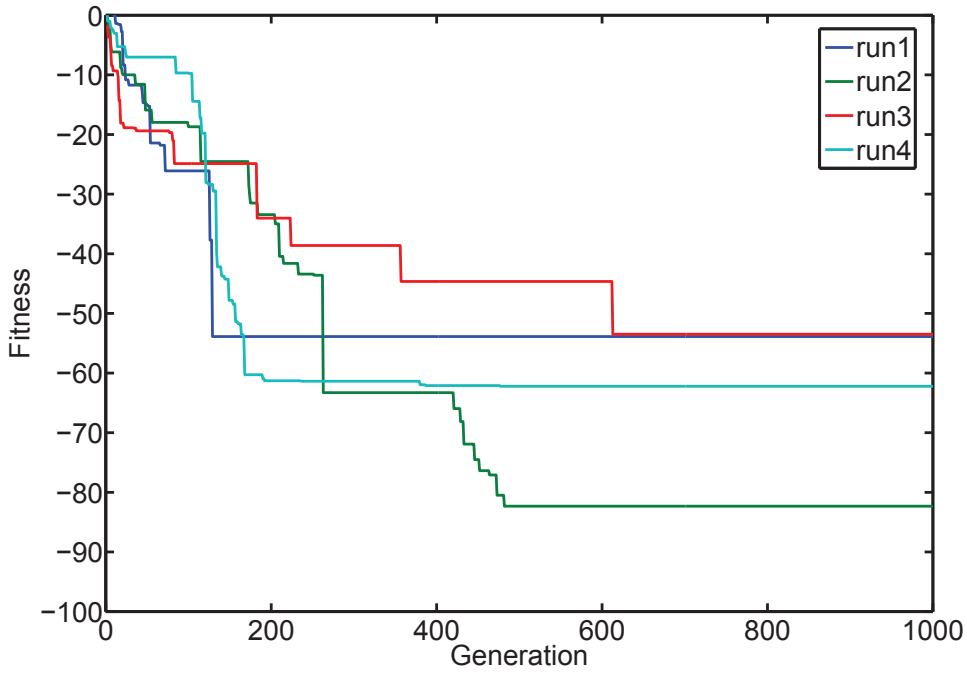


Figure 11.3: Fitness curves to evolve swimming individuals, their movements are controlled by CPGs.

CPGs is much lower than the frequencies of the sinuses in the previous chapter, the evolved swimming distances are hardly comparable. Run 2 is analyzed in more detail in the next section.

11.4.1 Analysis of Run 2

The fitness curve and the morphologies of some individuals are shown in Figure 11.4. An elongated shape develops quickly (generation 90), and subsequently the shape smoothens in later generations. The number of the CPGs also increases and their positions change.

Figure 11.5 shows the development of the best individual of run 2, while Figure 11.6 shows its swimming behavior. Most cells first divide, transform to a CPG and die afterwards. Because of the neurons on one side of the individual, the springs on this side do not change their natural length and the movement of the individual is only caused by the springs on the other side of the individual. At the end of the individual a triangle forms which has the appearance and seems to fulfill the function of a tail fin, as shown in Figure 11.7. It is also interesting that the resulting individual is asymmetric, contrary to the results of Jones et al. that show the advantage of symmetric morphologies [68].

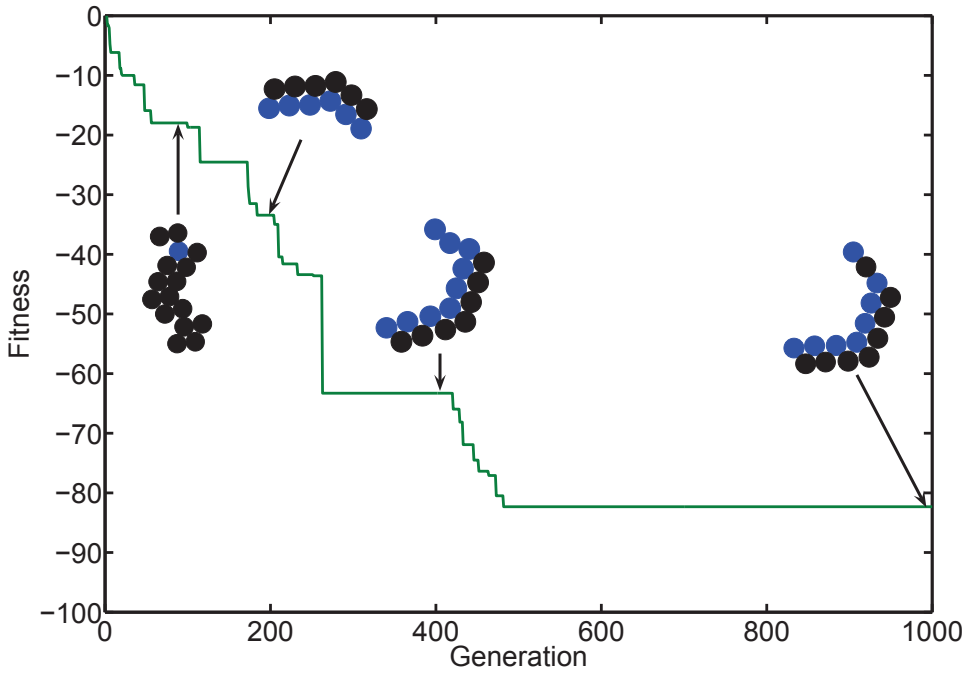


Figure 11.4: Fitness curve of run 2. The morphologies of the best individuals of generation 90, 200, 400 and 999 are drawn.

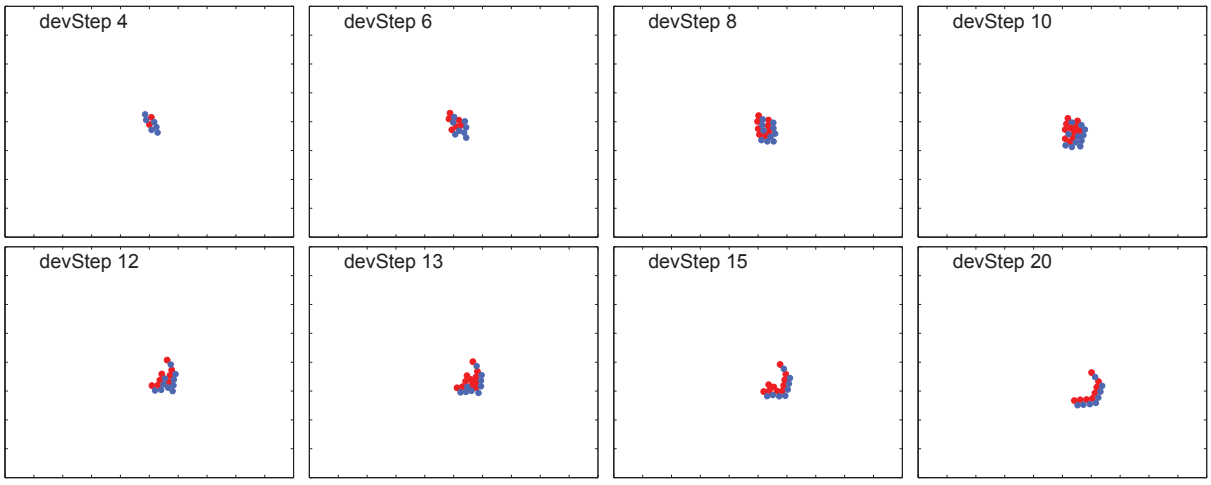


Figure 11.5: The development of the best individual of run 2 at the end of the evolution. Blue cells will divide in the following timestep, red cells transform to a CPG and will die in the following timestep.

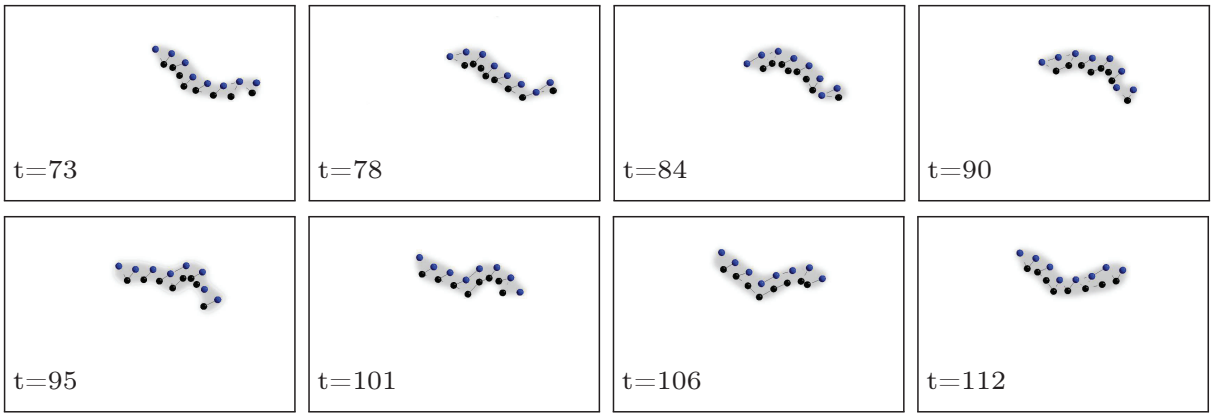


Figure 11.6: Swimming behavior of the best individual of run 2. CPGs are blue, all other cells are black.

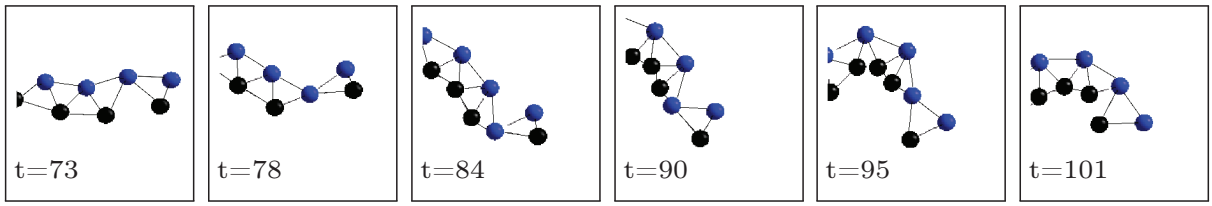
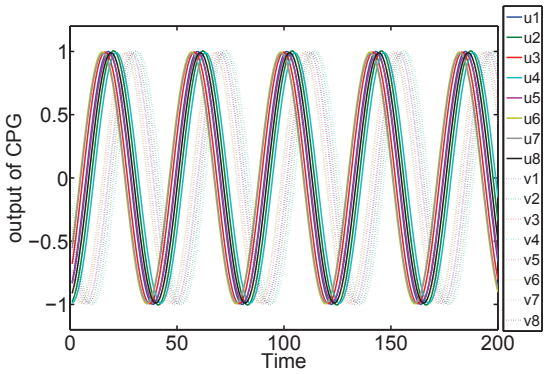
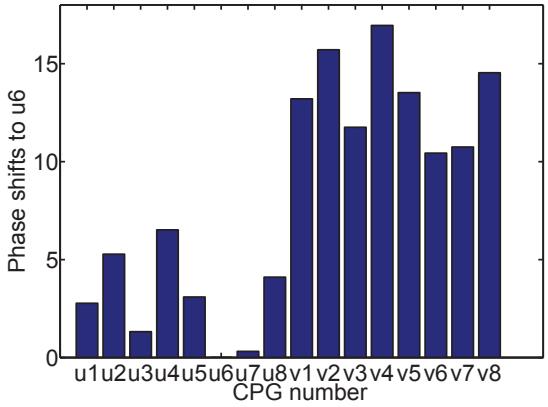


Figure 11.7: Tail fin of the best individual of run 2. CPGs are blue, all other cells are black.



(a) Time series of the different neurons.



(b) Phase shifts between the sinus curves of the different neurons relative to u6.

Figure 11.8: Output of all CPGs from the best individual of run 2.

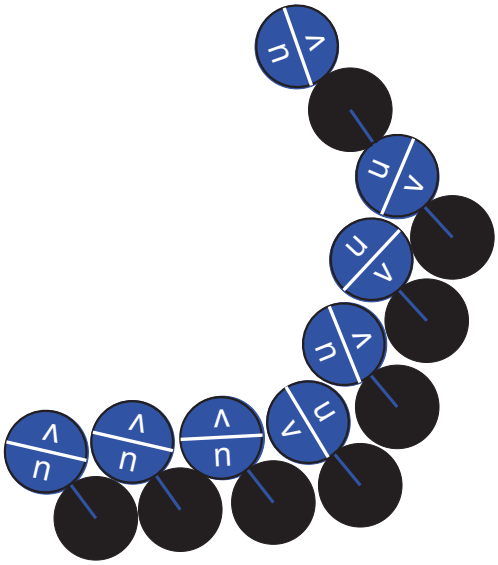


Figure 11.9: Orientation of the CPGs of the best individual of run 2. Since some CPGs are turned around, some muscle cells are connected to u and some to v, which causes the large phase shifts between the contractions of the springs.

The output of the CPGs are plotted in Figure 11.8, which shows that the phase shifts between the different CPGs are small. Figure 11.9 shows the orientations of the CPGs and we can see that some CPGs are turned around which results in a larger phase shift for the muscle cells.

11.5 Discussion

In this chapter, both the morphology and the control of the individuals are defined in one genome. The evolved individuals swim fast and perform plausible movements. Central pattern generators provide the ability to obtain an oscillating pattern with only a few neurons without limiting the connections or requiring long learning phases.

In comparison to the results of the previous chapter, smaller individuals evolve, which can be a result that no desired morphology of the individuals is defined in the fitness. However most individuals are elongated, which is influenced by the possible types of movements (which are by definition sinusoidal with phase shifts).

The definition of the morphology and control in one genome and the optimization of both at once is an advantage of this model compared with

most other models. E.g. Bongard showed that robots, that change their morphology in early stages of the evolution evolve faster and more robust than robots that never change their shape. The model developed here provides a good trade-off between computational costs and a biologically plausible encoding. Therefore its encoding is biologically more plausible than most other models. The usage of CPGs is also a step to biologically more plausible models compared to defining frequencies and phase shifts as in the previous chapter.

This model is to my knowledge the only model which combines the encoding of shape and control in one developmental process. This provides a lot of opportunities to analyze GRNs. It is hopefully possible with this encoding to find mechanisms of coevolution and to use these mechanisms to ease the optimization of both, form and functionality concurrently.

12 Summary and Outlook

Evolution and development are two of the most fundamental elements of nature. Simulating and analyzing these processes helps on the one hand to understand principles in biology and on the other hand this knowledge can be used to improve engineering design. Consequently, the objective of the work presented here was both to gain knowledge about developmental processes, and also to use this knowledge to build animats, which are artificial animals (physical robots in virtual simulations).

The core of this thesis is a model of gene regulatory networks, which is used to control the development of cells. Its level of abstraction is a trade-off between biological plausibility and the computational resources available. The developmental process starts with one cell in the center of a computation area and the cells can divide or die for several steps, so different individuals with e.g. different shapes can grow.

In the first part of this thesis, I analyzed gene regulatory networks (GRNs) and their evolution using the example of an elongated morphology. Firstly, stable development and regenerating capabilities were achieved, secondly, the occurrence of network motifs and their changes during the evolution were evaluated. A lot of small, frequently occurring network structures, the network motifs, are defined by biologists and seem to be important for the biological development. Thirdly, redundancy in the encoding and its influence on the design process was studied. In the second part, I presented a model for the development of morphology and control, where the control part was realized on different levels of abstraction.

In Chapter 5 I introduced a basic setup for the development of morphologies using the gene regulatory network model defined in Chapter 4. Three analyzes based on this experiment were conducted.

The first part was mostly concerned with developmental stability and regeneration. Evolution achieved a stable development, i.e. the morphology did not change any further in future developmental steps. In a second step, the individuals were injured, i.e. some cells were artificially deleted, and after five developmental steps, the shape was able to regrow. Most

other approaches use cells that are fixed on a grid or they use contact inhibition and therefore, the cells stop dividing when they are surrounded by other cells. In this framework, these cells can still divide. Due to their mechanical interactions, the new cell pushes the other cells outwards. This allows dynamic stability where during the whole development cells divide and die, but the overall shape remains constant. This process is in principle similar to the development of the cnidarian *Hydra*, which constantly renews its cells during its whole lifetime.

In Chapter 7, the network motifs of one basic run were analyzed. Most approaches in the literature count the static motifs, which are all possible motifs in a GRN. In comparison to these approaches, dynamic motifs were evaluated here, which means only the connections that were really used were counted. A general trend is that the overall number of motifs increases significantly at the beginning of evolution. A lot of changes in the number of motifs during the evolution were found with only small variations in the genome size. It seems that it is not just the increase of genetic material that is important for the evolutionary process, but of structured genetic material e.g. the dynamic network motifs. Motifs that do not influence the development were lost again during evolution. Therefore, it seems that the occurrence of motifs is under selectional control and that the increase of dynamic network motifs is related to the evolvability of the process.

This was followed by studying redundancy in the GRN during evolution. I found that removing redundancy from the genome during evolution decreases its performance. Furthermore, longer initial genomes, resulting in more genetic redundancy also improved the evolution, although there was an upper bound. An experiment with only a few and small genes performed the worst, which confirmed the hypothesis that redundancy is important for the evolution of this encoding method. Defining and analyzing different types of redundancy, e.g. structural and functional redundancy showed that they stay on a constant level. After removing redundant genes after a certain generation, the ongoing evolutionary process tends to re-insert redundancy in the genome in later generations. A new measure, the functional proximity, was introduced. It indicates the probability of a redundant gene to mutate to a functional gene. An increase of functional proximity was found in almost all successful runs, but not in the failed runs. Functional proximity could be used to predict whether an optimization is likely to find better solutions in future generations. A general conclusion is that to a certain degree redundancy creates oppor-

tunity for the evolutionary process. However, redundant genes must still be accessible thus functional proximity must be high.

In the second part of this thesis, I evolved animats on different levels of abstraction that can swim or catch food. Whereas the chapters of the first part dealt with evolving a morphology, the second part concentrated more on the evolution of a control part and the combination of morphology and control.

In Chapter 9, spiking neural networks were evolved to fulfill a food catching task, where the morphology was fixed. It was possible to evolve a developmental order, so that cell division, cell migration and neuron formation were executed in a sequence. In a second step the neuronal weights were evolved so that the individual was able to catch falling food.

I evolved a morphology with a simple control structure to fulfill a swimming task in Chapter 10. In this step, the morphology and the control were evolved in two separated genomes. Different morphologies with qualitatively different swimming strategies evolved, e.g. undulatory and peristaltic movements.

In the final chapter, a combined genome for morphology and control was used to fulfill the same swimming task. A different structural unit was used, so that cells were able to differentiate to central pattern generators that control the movements of the individual. Using this encoding, the GRN determines the development of the morphology and the control structure at once and in one genome.

The task of the individuals, e.g. swimming or food catching, is of course secondary, the possibility to evolve both a morphology and a control part together is important. The difference between encoding the morphology and control together or separately is visualized in Figure 1.1. One major contribution of this work is the improvement of the design process to build animats using one encoding for both the body and the control part.

This combined definition of morphology and control in one genome and the optimization of both concurrently is an advantage of this model compared with most other models, e.g. the models of Sims [120], Miconi and Channon [98] or Spector et al. [121] which use blocks as single elements of the individuals. The usage of cells, as provided here, can form shapes that are biologically more plausible. Their encoding of morphology and control is separated. Bongard evolves morphological units that are connected with joints using a development controlled by a GRN [20]. This promising model is used for object grasping and forward locomotion and

is less biologically plausible than the one presented here. The model developed for this thesis defines the morphology and control in one genome and therefore provides a good trade-off between computational costs and a biologically inspired encoding. Using such a biologically inspired model provides the opportunity to use other principles from nature to improve a design process. The usage of CPGs is advantageous compared to defining frequencies and phase shifts as in the work of Hornby and Pollack [60].

The results of both parts of this thesis have the potential to improve the design process. In the first part of the thesis, improvements of the developmental process and the evolution of the gene regulatory networks were achieved, whereas in the second part the encoding was modified, so that both the shape and the control part of an individual can be evolved together.

Design processes can be improved by generating the shape and control of a robot or any adaptive structure concurrently. An optimal shape can reduce the complexity of the control part, e.g. a passive dynamic walker can walk down a slope without control only because of their mechanical configuration, which means they are capable of stable, human-like walking [94]. In a standard design process, first the morphology of a robot is fixed and then the control is defined. Optimizing the morphology and the control concurrently provides the opportunity to find the optimal trade-off between both.

The GRN model developed in this thesis provides the concept of modularization by definition, since the cells are identical modules. The overall functionality is then determined by the connections between the modules and the evolution of the GRNs offers a method to create the functionality of the resulting individual. This decentralized control is the key to robustness.

A long-term goal for the approach presented in this thesis is to design robots and physically build them. Using such a method to build real robots and therefore to be able to compare it to standard approaches for real robots is still a long way away. There are a lot of open questions e.g. the single units (here: cells) need to be built in real hardware. Some promising approaches exist, e.g. by Meng et al. [96] or Lipson and Pollack [90], but a less biologically plausible or less complex encoding model is used.

The model as presented in this thesis is also useful to optimize topologies. Using different cell types, resulting in different materials can be used for structural optimization (e.g. Steiner et al. evolved lightweight and stable

structures [127]). New innovations in rapid prototyping, a process where 3D structures can be produced by e.g. a 3D printer, allow the building of complex structures and are likely to become cheaper in the future. It will also be possible to additionally print the control elements and therefore distribute them over the whole robot or structure in general. The 3D printer can then be used to build complex parts not only for testing but also to manufacture them on larger scales, termed adaptive manufacturing. This process would allow the use of extremely complex structures in products. The system described in this thesis could be used to optimize such complex structures.

A Discretization of the Mechanics of the Cells

In this appendix the details of the mechanical interactions between the cells are described. The mechanical interactions are computed at the end of each developmental step which is after the deletion and division of the cells. Overlapping cells push each other, nearby non overlapping cells adhere to each other and cells with a larger distance do not effect each other. This appendix describes the equations in more detail.

The force function $f(d)$ (see Chapter 4.3, Figure 4.4 and Equation 4.18) defines the absolute value of the force vector. It can be divided into x and y direction depending on the positions of the two cells $\boldsymbol{x}_i = (x_i, y_i)^T$ and $\boldsymbol{x}_j = (x_j, y_j)^T$:

$$\boldsymbol{f}^{i,j} = \frac{\boldsymbol{x}_i - \boldsymbol{x}_j}{d} \cdot f(d) = -\boldsymbol{f}^{j,i} \tag{A.1}$$

The position of a cell is updated depending on the sum of all forces of this cell and a small random noise (z):

$$\boldsymbol{F}^j = \sum_{i=1}^{N} \boldsymbol{f}^{i,j} + \boldsymbol{z}_j, \tag{A.2}$$

$$\boldsymbol{z}_j = (z_{j1}, z_{j2})^T, z_{ji} \sim \mathcal{U}(-10^{-3}, 10^{-3}), i \in \{1,2\}, \tag{A.3}$$

where N is the number of cells. The change in the velocity ($\boldsymbol{v}$) and position of each cell is

$$\begin{aligned} \boldsymbol{\Delta x}(t) &= \begin{cases} \frac{\boldsymbol{v}(t-1)\Delta t}{|\boldsymbol{v}(t-1)\Delta t|}\Delta x_{max} & \text{if } |\boldsymbol{v}(t-1)\Delta t| < \Delta x_{max} \\ \boldsymbol{v}(t-1)\Delta t & \text{otherwise} \end{cases} \\ \boldsymbol{\Delta v}(t) &= (1-c)\boldsymbol{v}(t-1) + \frac{\Delta t}{m}\boldsymbol{F}(t). \end{aligned} \tag{A.4}$$

The forces are computed until all cells do not move any more ($|\boldsymbol{\Delta x_i}| < \Delta x_{min} \forall i \in [1,N]$) or ($|F| < F_{min} \forall i \in [1,N]$) or a maximum number of iterations is reached. The constants are listed in Table A.1.

Table A.1: Constants of the computation of the mechanics between the cells.

Δx_{max}	1
Δx_{min}	0.1
F_{min}	0.05
maximum number of iterations	900
$\boldsymbol{v}(0)$	$(0,0)^T$
mechanical damping c	0.95
mass of each cell m	1

B Fitness Curves of all Experiments for the Analysis of Redundancy

The experiments associated to the fitness curves shown in Figures B.1-B.6 are described in Chapter 8 and the definitions of the different setups are given in Table 8.1.

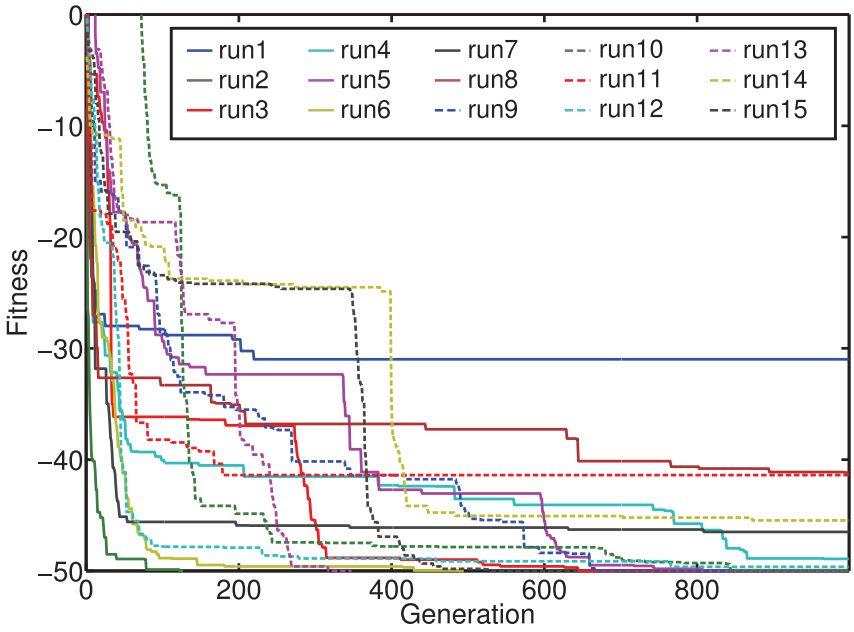


Figure B.1: Fitness curves of setup 2.

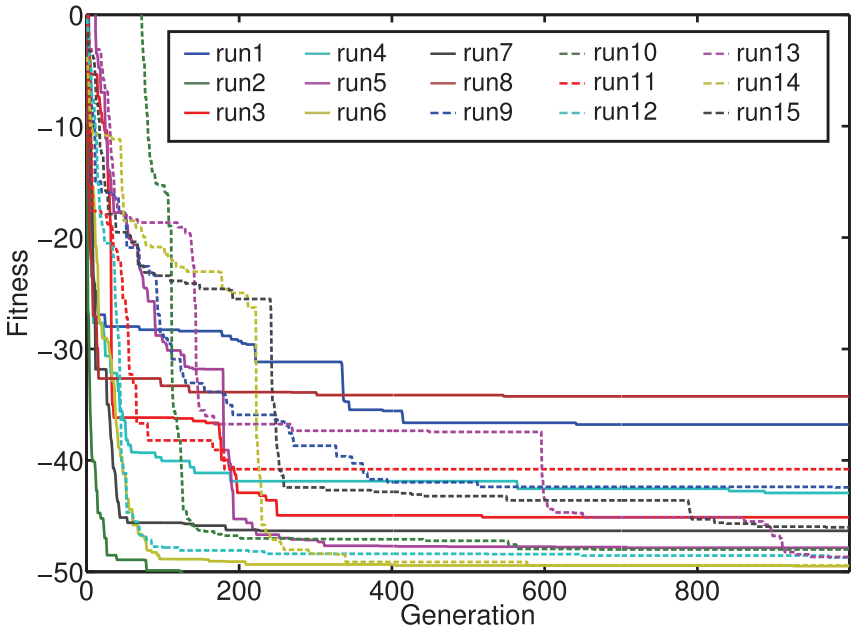


Figure B.2: Fitness curves of setup 3.

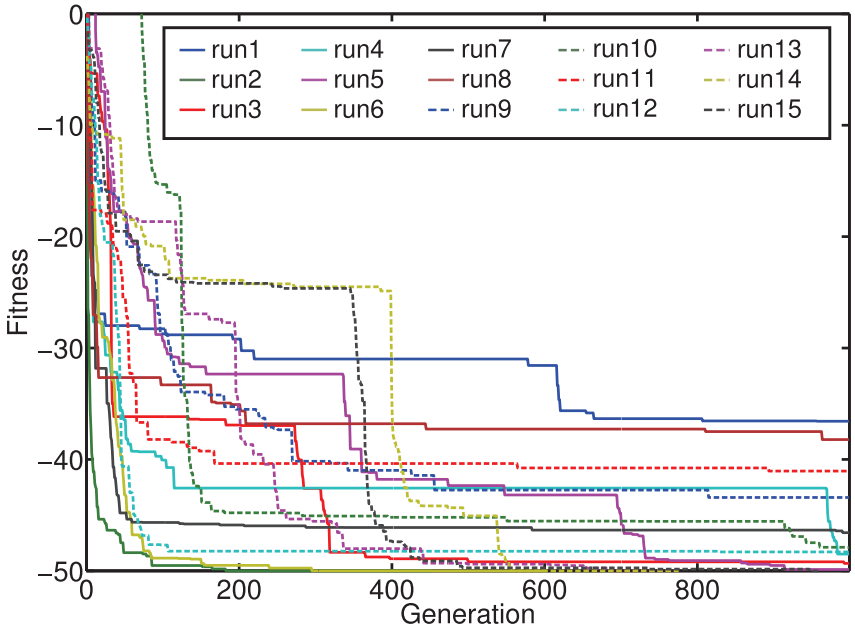


Figure B.3: Fitness curves of setup 5.

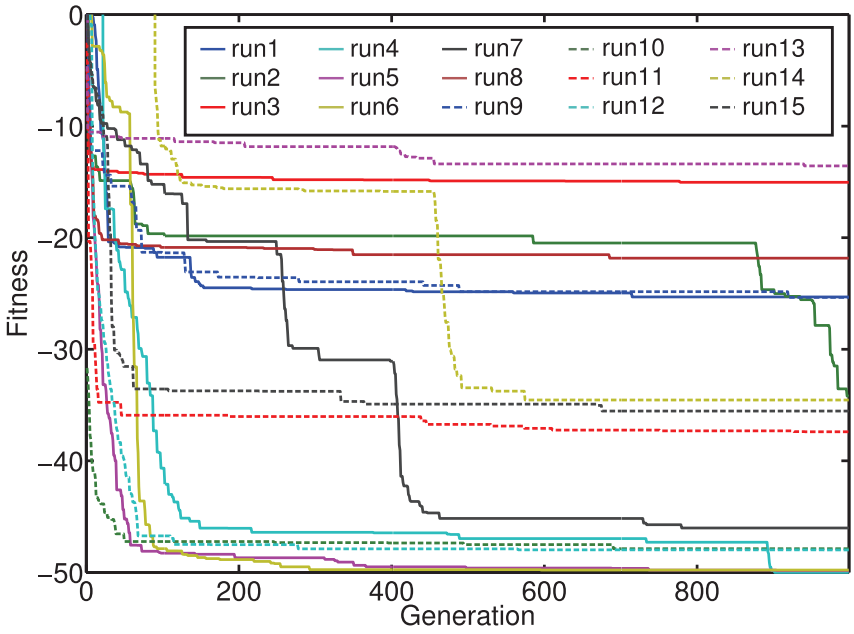


Figure B.4: Fitness curves of setup 6.

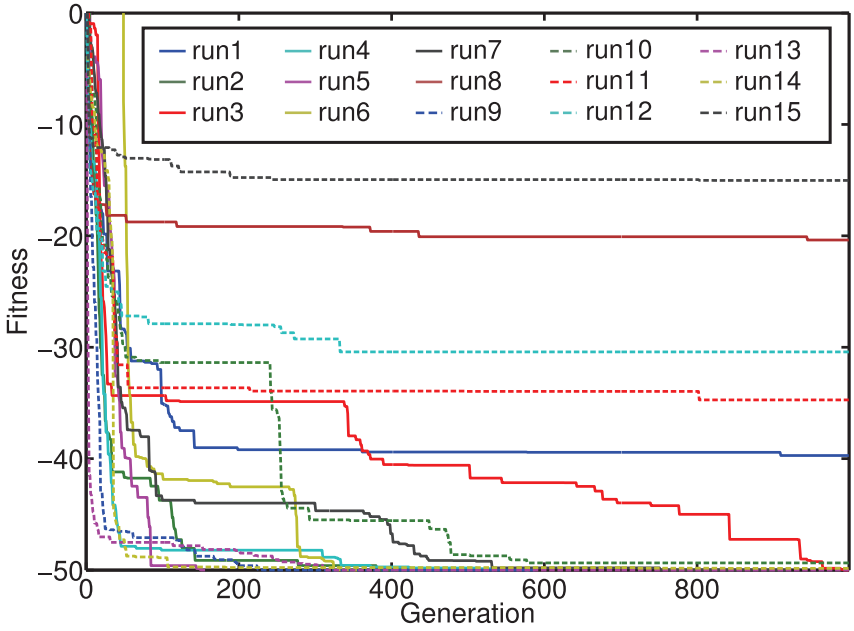


Figure B.5: Fitness curves of setup 8.

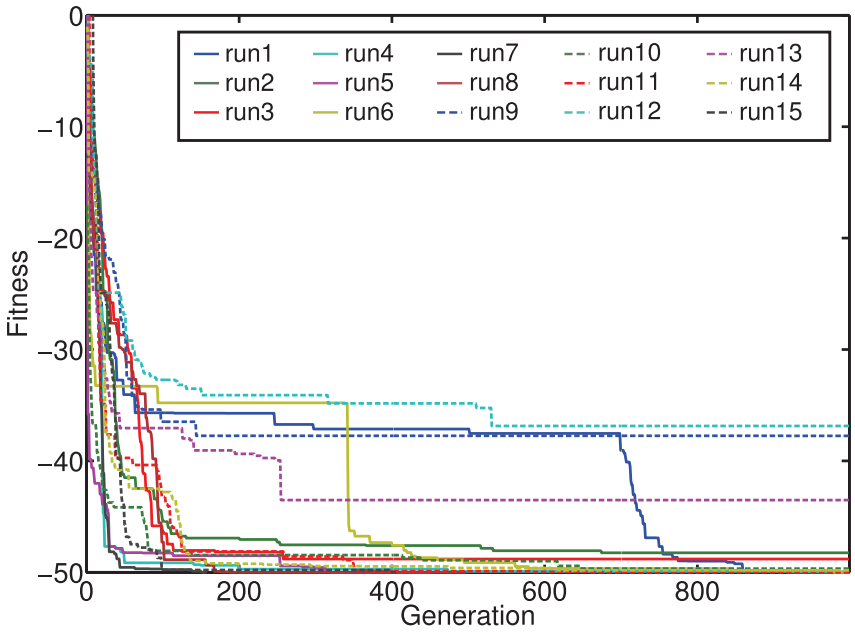


Figure B.6: Fitness curves of setup 9.

C Simulation of the Swimming Animats

To simulate the physical interactions of a swimming animat, a computation environment for the cells and connections between the cells and external water forces are necessary. The next section describes the simulation environment breve [80] which is used for the experiments described in this thesis. This description is followed by a section about the self programmed water forces.

C.1 The Simulation Environment Breve

The simulation environment breve is a free, open-source software package which performs 3D simulations [80]. Different simple shapes, joints and springs can be coupled and their movements can be computed [79]. It is possible to simulate internal and external forces, make a rigid body simulation, detect collisions, response to collisions, have static and dynamic friction, and much more.

Breve has a Python^{TM} interface and is object-oriented. It has an OpenGL display engine to visualize the movements.

There are no external water forces provided in breve, so they are programmed according to the equations in the next section.

C.2 Simulation of Water Forces

Sfakiotakis and Tsakiris present the simulation environment SIMUUN for undulatory locomotion [117]. They use a simplified model for the water forces, which is used with some changes for the work presented in this thesis. The assumptions and the resulting equations they use are described in the following:

- The Reynolds number is high enough for inertial forces to dominate over viscous forces ($400 < Re < 4 \cdot 10^5$). This assumption is only

valid for inviscous swimming, so it cannot be applied to undulatory swimming of microorganisms.

- Stationarity, which means that the force of the fluid on a single link is only because of the motion of this link.
- The forces for tangential and normal directions are decoupled.

So the Navier-Stokes equations can be simplified to:

$$F_T^i = -\lambda_T^i sgn(v_T^i) \cdot (v_T^i)^2 \tag{C.1}$$

$$F_N^i = -\lambda_N^i sgn(v_N^i) \cdot (v_N^i)^2. \tag{C.2}$$

$$\lambda^i = \frac{1}{2}\rho C^i S^i, \tag{C.3}$$

which means, the model ignores secondary effects of water movement. S is the effective area of the link i, ρ is the fluid density and C a shape coefficient. $\lambda_N/\lambda_T \simeq 10$ was measured for a swimming grass snake. The ratio of λ_N/λ_T defines the possibility and the direction of the movement of the animal depending on the undulatory wave of its body. The values for λ and therefore the values for S and C used in this work are found through trial and error to achieve fast movements.

List of Publications

- Till Steiner, Lisa Schramm, Yaochu Jin, and Bernhard Sendhoff. Emergence of feedback in artificial gene regulatory networks. In *Proceedings of the IEEE Conference on Evolutionary Computation (CEC)*, pages 867–874, Singapore, 2007

- Yaochu Jin, Lisa Schramm, and Bernhard Sendhoff. A gene regulatory model for the development of primitive nervous systems. In *Proceedings of the 15th International Conference on Neural Information Processing (ICONIP)*, volume 5506/2009, pages 48–55, Auckland, New Zealand, 2008

- Lisa Schramm, Yaochu Jin, and Bernhard Sendhoff. Emerged coupling of motor control and morphological development in evolution of multi-cellular animats. In *Proceedings of the European Conference on Artificial Life (ECAL)*, pages 25–32, Budapest, Hungary, 2009

- Till Steiner, Yaochu Jin, Lisa Schramm, and Bernhard Sendhoff. Dynamic links and evolutionary history in simulated gene regulatory networks. In Sanjoy Das, Doina Caragea, Stephen Welch, and William H. Hsu, editors, *Handbook of Research on Computational Methodologies in Gene Regulatory Networks*, pages 498–522. IGI Publishing, 2009

- Lisa Schramm, Vander Valente Martins, Yaochu Jin, and Bernhard Sendhoff. Analysis of gene regulatory network motifs in evolutionary development of multicellular organisms. In *Proceedings of the 12th International Conference on the Synthesis and Simulation of Living Systems (ALife XII)*, pages 133–140, Odense, Denmark, 2010

- Lisa Schramm, Yaochu Jin, and Bernhard Sendhoff. Redundancy creates opportunity in developmental representations. In *Proceedings of the IEEE Symposium Series on Computational Intelligence 2011 (IEEE ALIFE 2011)*, pages 203–210, Paris, France, 2011

- Lisa Schramm and Bernhard Sendhoff. An animat's cell doctrine. In *Proceedings of the European Conference on Artificial Life (ECAL)*, pages 739–746, Paris, France, 2011

Bibliography

[1] Sophie Abby and Vincent Daubin. Comparative genomics and the evolution of prokaryotes. *Trends in Microbiology*, 15(3):135–141, 2007.

[2] Bruce Alberts, Alexander Johnson, Julian Lewis, Martin Raff, Keith Roberts, and Peter Walter. *Molecular Biology of the Cell, Fourth Edition*. Garland, 2002.

[3] Uri Alon. *An Introduction to Systems Biology: Design Principles of Biological Circuits*. Chapman & Hall, July 2006.

[4] Uri Alon. Network motifs: theory and experimental approaches. *Nature Reviews Genetics*, 8:450–461, 2007.

[5] Tim Andersen, Richard Newman, and Tim Otter. Shape homeostasis in virtual embryos. *Artificial Life*, 15(2):161–183, 2009.

[6] Hilary L. Ashe and James Briscoe. The interpretation of morphogen gradients. *Development*, 133(3):385–394, 2006.

[7] Daniel Ashlock. *Evolutionary Computation for Modeling and Optimization*. Springer New York, 2006.

[8] M. Madan Babu, Nicholas M. Luscombe, L. Aravind, Mark Gerstein, and Sarah A. Teichmann. Structure and evolution of transcriptional regulatory networks. *Current Opinion in Structural Biology*, 14(3): 283–291, 2004.

[9] Thomas Bäck. *Evolutionary Algorithms in Theory and Practice: Evolution Strategies, Evolutionary Programming, Genetic Algorithms*. Oxford Univ. Press., 1996.

[10] Wolfgang Banzhaf. Genotype-phenotype-mapping and neutral variation - a case study in genetic programming. In *Parallel Problem Solving from Nature - PPSN III*, pages 322–332, 2006.

[11] Albert-László Barabási and Eric Bonabeau. Scale-free networks. *Scientific American*, 288(5):50–59, 2003.

[12] David Basanta, Mark Miodownik, and Buzz Baum. The evolution of robust development and homeostasis in artificial organisms. *PLoS Computational Biology*, 4(3):e1000030, 03 2008.

[13] Randall D. Beer. *Beyond Control: The Dynamics of Brain-Body-Environment Interaction in Motor Systems*, pages 7–24. Springer, 2009.

[14] Peter J. Bentley and Sanjeev Kumar. Three ways to grow designs: A comparison of embryogenies for an evolutionary design problem. In *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO '99)*, pages 35–43, Orlando, Florida, USA, 1999.

[15] Michael J. Benton. The red queen and the court jester: Species diversity and the role of biotic and abiotic factors through time. *Science*, 323(5915):728–732, 2009.

[16] Stefano Berri, Jordan H. Boyle, Manlio Tassieri, Ian A. Hope, and Netta Cohen. Forward locomotion of the nematode C. elegans is achieved through modulation of a single gait. *HFSP Journal*, 3: 186–193, 2009.

[17] Hans-Georg Beyer and Hans-Paul Schwefel. Evolution strategies –a comprehensive introduction. *Natural Computing: an international journal*, 1(1):3–52, 2002. ISSN 1567-7818.

[18] Hans R. Bode. The interstitial cell lineage of hydra: a stem cell system that arose early in evolution. *Journal of Cell Science*, 109: 1155–1164, 1996.

[19] Hans R. Bode. Head regeneration in *Hydra*. *Developmental Dynamics*, 226(2):225–236, 2003.

[20] Josh Bongard. *Incremental Approaches to the Combined Evolution of a Robot's Body and Brain*. PhD thesis, University of Birmingham, 2003.

[21] Josh Bongard. Morphological change in machines accelerates the evolution of robust behavior. *Proceedings of the National Academy of Sciences*, 108(4):1234–1239, 2011.

[22] Josh C. Bongard and Chandana Paul. Investigating morphological symmetry and locomotive efficiency using virtual embodied evolution. In *From Animals to Animats: Proceedings of the 6th International Conference on the Simulation of Adaptive Behaviour*, pages 420–429. MIT Press, 2000.

[23] Olaf Breidbach and Wolfram Kutsch, editors. *The Nervous System of Invertebrates: An Evolutionary and Comparative Approach.* Birkhäuser Verlag, 1995.

[24] Angelo Cangelosi, Domenico Parisi, and Stefano Nolfi. Cell division and migration in a 'genotype'for neural networks. *Networks - Computational in Neural Systems*, 5:479–515, 1994.

[25] Soon-Jo Chung and Jean-Jacques Slotine. On synchronization of coupled hopf-kuramoto oscillators with phase delays. *submitted to IEEE Transactions on Automatic Control*, 2010. arXiv:1004.5366v1.

[26] Stefano Ciliberti, Olivier C. Martin, and Andreas Wagner. Robustness can evolve gradually in complex regulatory gene networks with varying topology. *PLoS Comput Biol*, 3(2):e15, 02 2007. doi: 10.1371/journal.pcbi.0030015.

[27] Marco Colasanti, Giorgio Venturini, Angelo Merante, Giovanni Musci, and Giuliana M. Lauro. Nitric oxide involvement in hydra vulgaris very primitive olfactory-like system. *The Journal of Neuroscience*, 17(1):493–499, 1997.

[28] Max B. Cooper, Matthew Loose, and John F.Y. Brookfield. Evolutionary modelling of feed forward loops in gene regulatory networks. *Biosystems*, 91(1):231–244, 2008.

[29] Anton Crombach and Paulien Hogeweg. Evolution of evolvability in gene regulatory networks. *PLoS Comput Biol*, 4(7):e1000112, 07 2008.

[30] Charles Darwin. *On the Origin of Species by Means of Natural Selection.* J. Murray, London, 1859.

[31] Charles N. David, Nikola Schmidt, Marsha Schade, Barbara Pauly, Olga Alexandrova, and Angelika Böttger. Hydra and the evolution of apoptosis. *Integrative and Comparative Biology*, 45:631–638, 2005.

[32] Peter Dayan and Laurence F. Abbott. *Theoretical Neuroscience: Computational and Mathematical Modeling of Neural Systems*. The MIT Press, 2001.

[33] Hidde de Jong. Modeling and simulation of genetic regulatory systems: A literature review. *Journal of Computational Biology*, 9(1): 67–103, 2002.

[34] Frank Dellaert and Randall D. Beer. Toward an evolvable model of development for autonomous agent synthesis. In *Artificial Life IV*, pages 246–257, 1994.

[35] Markus Diesmann and Marc-Oliver Gewaltig. NEST: An environment for neural systems simulations. In Theo Plessner and Volker Macho, editors, *Forschung und wissenschaftliches Rechnen*, volume 58, Göttingen, 2003. Gesellschaft für wissenschaftliche Datenverarbeitung.

[36] Keith L. Downing. Supplementing evolutionary developmental systems with abstract models of neurogenesis. In *Genetic and Evolutionary Computation Conference*, pages 990–996, 2007.

[37] Marc Ebner, Patrick Langguth, J. Albert, Mark Schakleton, and Rob Shipman. On neutral networks and evolvability. In *Congress on Evolutionary Computation*, pages 27–30. IEEE Press, 2001.

[38] Peter Eggenberger. Evolving morphologies of simulated 3d organisms based on differential gene expression. In *Proceedings of the 4th European Conference on Artificial Life (ECAL)*, pages 205–213. MIT Press, 1997.

[39] Peter Eggenberger Hotz. Asymmetric cell division and its integration with other developmental processes for artificial evolutionary systems. In *Proceedings of the Ninth International Conference on the Simulation and Synthesis of Living Systems*, pages 387–398. MIT Press, 2004.

[40] Peter Eggenberger Hotz, Gabriel Gómez, and Rolf Pfeifer. Evolving the morphology of a neural network for controlling a foveating retina - and its test on a real robot. In *Proceedings of the 8th International Conference on Artificial Life (ALife 03)*, pages 243–251. MIT Press, 2003.

[41] Diego Federici. Evolving developing spiking neural networks. In *Congress on Evolutionary Computation*, pages 543–550, 2005.

[42] Diego Federici and Keith Downing. Evolution and development of a multicellular organism: Scalability, resilience, and neutral complexification. *Artificial Life*, 12(3):381–409, 2006. doi: 10.1162/artl.2006.12.3.381. URL `http://www.mitpressjournals.org/doi/abs/10.1162/artl.2006.12.3.381`.

[43] Kurt Fleischer. Investigations with a multicellular develpmental model. In *Proceedings of the 5th International Workshop on the Synthesis and Simulation of Living Systems (ALife V)*, pages 229–236, 1996.

[44] Kurt Fleischer and Alan H. Barr. A simulation testbed for the study of multicellular development: The multiple mechanisms of morphogenesis. In *Artificial Life III: Proceedings of the Workshop on Artificial Life*, pages 389–416, 1993.

[45] Paul François and Vincent Hakim. Design of genetic networks with specified functions by evolution in silico. *Proceedings of the National Academy of Sciences of the United States of America*, 101(2):580–585, 2004.

[46] Douglas J. Futuyma. *Evolutionary Biology*. Sinauer Associates Inc., 1998.

[47] Nicholas Geard and Kai Willadsen. Dynamical approaches to modeling developmental gene regulatory networks. *Birth Defects Research Part C*, 87(2):131–142, 2009.

[48] John Gerhart and Marc Kirschner. *Cells, Embryos, and Evolution. Towards a Cellular and Developmental Understanding of Phenotypic Variation and Evolutionary Adaptability*. Blackwell, 1997.

[49] Marc-Oliver Gewaltig and Markus Diesmann. NEST (Neural Simulation Tool). *Scholarpedia*, 2(4):1430, 2007. Source code available at: http://www.nest-initiative.org/.

[50] Jean D Gibbons. *Nonparametric Statistical Inference*. Marcel Dekker, 1985.

[51] Scott F. Gilbert. *Devlopmental Biology*. Sinauer Associates, Inc., 2003.

[52] Frédéric Gruau. *Neural Network Synthesis using Cellular Encoding and the Genetic Algorithm.* PhD thesis, Ecole Normale Supérieure de Lyon and Centre d'études nucléaire de Grenoble, France, 1994.

[53] Simon Harding and Wolfgang Banzhaf. Artificial development. In Rolf W. Würz, editor, *Organic Computing*, chapter 9, pages 201–219. Springer, 2008.

[54] Simon Harding and Julian F. Miller. The dead state: A comparison between developmental and direct encodings. In *Genetic and Evolutionary Computation Conference (GECCO2006) Workshop Program: Complexity through Development and Self-Organizing Representations (CODESOAR)*, 2006.

[55] Jeff Hasty, David McMillen, Farren Isaacs, and James J. Collins. Computational studies of gene regulatory networks: in numero molecular biology. *Nature Reviews Genetics*, 2:268–279, 2001.

[56] Jonathan D. Hiller and Hod Lipson. Evolving amorphous robots. In *Proceedings of the 12th International Conference on the Synthesis and Simulation of Living Systems (ALife XII)*, pages 717–724, Odense, Denmark, 2010.

[57] Oliver Hobert. Specification of the nervous system. In *WormBook*. The C. elegans Research Community, 2005. doi: 10.1895/wormbook.1.12.1. http://www.wormbook.org.

[58] Jonathan Hodgkin. Introduction to genetics and genomics. In *WormBook*. The C. elegans Research Community, 2005. doi: 10.1895/wormbook.1.17.1. http://www.wormbook.org.

[59] Thomas Holstein, Bert Hobmayer, and Ulrich Technau. Cnidarians: An evolutionarily conserved model system for regeneration? *Developmental Dynamics*, 226:257–267, 2003.

[60] Gregory Hornby and Jordan Pollack. Evolving L-systems to generate virtual creatures. *Computers & Graphics*, 25:1040–1048, 2001.

[61] Christian Igel, Verena Heidrich-Meisner, and Tobias Glasmachers. Shark. *Journal of Machine Learning Research*, 9:993–996, 2008.

[62] Auke Jan Ijspeert and Jéróme Kodjabachian. Evolution and development of a central pattern generator for the swimming of a lamprey. *Artificial Life*, 5(3):247–269, 1999.

[63] Douglas H. Irvine and Michael A. Savageau. Efficient solution of nonlinear ordinary differential equations expressed in s-system canonical form. *SIAM Journal on Numerical Analysis*, 27(3):704–735, 1990.

[64] Yaochu Jin, Lisa Schramm, and Bernhard Sendhoff. A gene regulatory model for the development of primitive nervous systems. In *Proceedings of the 15th International Conference on Neural Information Processing (ICONIP)*, volume 5506/2009, pages 48–55, Auckland, New Zealand, 2008.

[65] Yaochu Jin, Robin Gruna, Ingo Paenke, and Bernhard Sendhoff. Evolutionary multi-objective optimization of robustness and innovation in redundant genetic representations. In *IEEE Symposium on Computational Intelligence in Multi-Criteria Decision-Making*, pages 38–45, 2009.

[66] Michał Joachimczak and Borys Wròbel. Evolution of the morphology and patterning of artificial embryos: scaling the tricolour problem to the third dimension. In *Proceedings of the European Conference on Artificial Life (ECAL)*, pages 33–40, 2009.

[67] Michał Joachimczak and Borys Wròbel. Complexity of the search space in a model of artificial evolution of gene regulatory networks controlling 3d multicellular morphogenesis. *Advances in Complex Systems*, 12:347–369, 2009.

[68] Ben Jones, Yaochu Jin, Bernhard Sendhoff, and Xin Yao. Evolving functional symmetry in a three dimensional model of an elongated organism. In *Proceedings of the 11th International Conference on the Simulation and Synthesis of Living Systems (ALife)*, pages 305–312. MIT Press, 2008.

[69] Ben H. Jones, Yaochu Jin, Xin Yao, and Bernhard Sendhoff. Evolution of neural organization in the hydramat. In *Proceedings of the 15th International Conference on Neural Information Processing (ICONIP)*, volume 5506/2009, pages 216–223, Auckland, New Zealand, 2008.

[70] Ben H. D. Jones. *The Evolutionary Emergence of Neural Organisation in Computational Models of Primitive Organisms*. PhD thesis, University of Birmingham, 2010.

[71] John H. Kaas, Georg F. Striedter, and John L. R. Rubenstein, editors. *Evolution of nervous systems*, volume 1. Elsevier, first edition, 2007.

[72] Henrik Kacser and Richard Beeby. Evolution of catalytic proteins or on the origin of enzyme species by means of natural selection. *J Mol Evol*, 20:38–51, 1984.

[73] Nadav Kashtan and Uri Alon. Spontaneous evolution of modularity and network motifs. *Proceedings of the National Academy of Sciences*, 102(39):13773–13778, 2005.

[74] Motoo Kimura. *The neutral theory of molecular evolution*. Cambridge University Press, 1983.

[75] Hiroaki Kitano. Designing neural networks using genetic algorithms with graph generation system. *Complex Systems Journal*, 4:461–476, 1990.

[76] Hiroaki Kitano. A simple model of neurogenesis and cell differentiation based on evolutionary large-scale chaos. *Artificial Life*, 2(1): 79–99, 1995.

[77] Hiroaki Kitano. Systems biology: A brief overview. *Science*, 295 (5560):1662–1664, 2002.

[78] Hiroaki Kitano. Computational systems biology. *Nature*, 420:206–210, 2002.

[79] Jon Klein. breve: a 3d simulation environment for the simulation of decentralized systems and artificial life. In *Proceedings of Artificial Life VIII, the 8th International Conference on the Simulation and Synthesis of Living Systems*. The MIT Press, 2002.

[80] Jon Klein. BREVE simulation environment, 2008. Source code available at: http://www.spiderland.org/.

[81] Johannes F. Knabe, Maria J. Schilstra, and Chrystopher Nehaniv. Evolution and morphogenesis of differentiated multicellular organisms: autonomously generated diffusion gradients for positional information. In *Proceedings of the 11th International Conference on the Simulation and Synthesis of Living Systems (ALife)*. MIT Press, 2008.

[82] Osamu Koizumi. Developmental neurobiology of hydra, a model animal of cnidarians. *Canadian Journal of Zoology*, 80(10):1678–1689, 2002.

[83] Osamu Koizumi. Nerve ring of the hypostome in hydra: Is it an origin of the central nervous system of bilaterian animals? *Brain, Behavior and Evolution*, 69(2):151–159, 2007.

[84] Maciej Komosiński and Adam Rotaru-Varga. Comparison of different genotype encodings for simulated three-dimensional agents. *Artificial Life*, 7(4):395–418, 2001.

[85] David C. Krakauer and Joshua B. Plotkin. Redundancy, antiredundancy, and the robustness of genomes. *Proceedings of the National Academy of Sciences*, 99(3):1405–1409, 2002.

[86] Sanjeev Kumar and Peter J. Bentley. Biologically inspired evolutionary development. In *Evolvable Systems: From Biology to Hardware*, pages 57–68, Trondheim, Norway, 2003.

[87] P. Dwight Kuo, Wolfgang Banzhaf, and André Leier. Network topology and the evolution of dynamics in an artificial genetic regulatory network model created by whole genome duplication and divergence. *BioSystems*, 85(3):177 – 200, 2006.

[88] Yung-Keun Kwon and Kwang-Hyun Cho. Quantitative analysis of robustness and fragility in biological networks based on feedback dynamics. *Bioinformatics*, 7(24):987–994, 2008.

[89] Aristid Lindenmayer and Grzegorz Rozenberg. Developmental systems and languages. In *STOC '72: Proceedings of the fourth annual ACM symposium on Theory of computing*, pages 214–221, New York, NY, USA, 1972. ACM.

[90] Hod Lipson and Jordan. B. Pollack. Automatic design and manufacture of robotic lifeforms. *Nature*, 406:974–978, 2000.

[91] Michael Lynch and John S. Conery. The evolutionary fate and consequences of duplicate genes. *Science*, 290:1151–1155, 2002.

[92] Wolfgang Maass. Networks of spiking neurons: The third generation of neural network models. In *Australian Conference on Neural Networks*, pages 1–10. Australian National University, 1996.

[93] Mariagiovanna Mazzapioda, Angelo Cangelosi, and Stefano Nolfi. Co-evolving morphology and control: A distributed approach. In *Proceedings of the IEE Conference on Evolutionary Computation (CEC)*, pages 2217–2224, 2009.

[94] Tad McGeer. Passive dynamic walking. *International Journal of Robotics Research*, 9(2):62–82, 1990.

[95] Hans Meinhard. A model for pattern formation of hypostome, tentacles and foot in Hydra: how to form structures close to each other, how to form them at a distance. *Developmental Biology*, 157(2): 321–333, 1993.

[96] Yan Meng, Yuyang Zhang, and Yaochu Jin. A morphogenetic approach to self-reconfigurable modular robots using a hybrid hierarchical gene regulatory network. In *Proceedings of the 12th International Conference on the Synthesis and Simulation of Living Systems (ALife XII)*, pages 765–772, Odense, Denmark, 2010.

[97] Thomas Mestl, Erik Plahte, and Stig W. Omholt. A mathematical framework for describing and analysing gene regulatory networks. *Journal of Theoretical Biology*, 176(2):291–300, 1995.

[98] Thomas Miconi and Alastair Channon. An improved system for artificial creatures evolution. In *Proceedings of Artificial Life X*, pages 255–261, 2006.

[99] Marijana Miljkovic-Licina, Dominique Gauchat, and Brigitte Galliot. Neuronal evolution: analysis of regulatory genes in a first-evolved nervous system, the hydra nervous system. *Biosystems*, 76 (1-3):75–87, 2004.

[100] Julian Francis Miller. Evolving a self-repairing, self-regulating, french flag organism. In *Proceedings of the GECCO*, pages 129–139, 2004.

[101] Renato E. Mirollo and Steven H. Strogatz. Synchronization of pulse-coupled biological oscillators. *SIAM Journal on Applied Mathematics*, 50(6):1645–1662, 1990.

[102] James D. Murray. *Mathematical Biology I: An Introduction*. Springer, third edition, 2008.

[103] Stefano Nolfi and Domenico Parisi. Evolving artificial neural networks that develop in time. In *European Conference on Artificial Life*, pages 353–367, 1995.

[104] Eilen Nordlie, Marc-Oliver Gewaltig, and Hans Ekkehard Plesser. Towards reproducible descriptions of neuronal network models. *PLoS Computational Biology*, 5(8):e1000456, 08 2009.

[105] Nicolas Oros, Volker Steuber, Neil Davey, Lola Cañamero, and Rod Adams. Evolution of bilateral symmetry in agents controlled by spiking neural networks. In *IEEE Symposium on Artificial Life*, pages 116–123. IEEE Press, 2009.

[106] Thomas Pfeiffer, Orkun S Soyer, and Sebastian Bonhoeffer. The evolution of connectivity in metabolic networks. *PLoS Biol*, 3(7): e228, 06 2005. doi: 10.1371/journal.pbio.0030228.

[107] Jonathan T. Pierce-Shimomura, Beth L. Chen, James J. Mun, Raymond Ho, Raman Sarkis, and Steven L. McIntire. Genetic analysis of crawling and swimming locomotory patterns in c. elegans. *PNAS*, 105(52):20982–20987, 2008.

[108] Sean Psujek and Randall D. Beer. Developmental bias in evolution: Evolutionary accessibility of phenotypes in a model of evo-devo systems. *Evolution and Development*, 10(3):375–390, 2008.

[109] Adrian P. Quayle and Seth Bullock. Modelling the evolution of genetic regulatory networks. *Journal of Theoretical Biology*, 238(4): 737–753, 2006.

[110] Berthold Ruf. *Computing and Learning with Spiking Neurons – Theory and Simulations*. PhD thesis, Technische Universität Graz, Austria, 1998.

[111] Anastasia A. Samsonova, Mahesan Niranjan, Steven Russell, and Alvis Brazma. Prediction of gene expression in embryonic structures of *Drosophila melanogaster*. *PLoS Comput Biol*, 3(7):e144, 07 2007.

[112] Dan H. Sanes, Thomas A. Reh, and William A. Harris. *Development of the Nervous System*. Academic Press, Amsterdam, 2006.

[113] Lisa Schramm and Bernhard Sendhoff. An animat's cell doctrine. In *Proceedings of the European Conference on Artificial Life (ECAL)*, pages 739–746, Paris, France, 2011.

[114] Lisa Schramm, Yaochu Jin, and Bernhard Sendhoff. Emerged coupling of motor control and morphological development in evolution of multi-cellular animats. In *Proceedings of the European Conference on Artificial Life (ECAL)*, pages 25–32, Budapest, Hungary, 2009.

[115] Lisa Schramm, Vander Valente Martins, Yaochu Jin, and Bernhard Sendhoff. Analysis of gene regulatory network motifs in evolutionary development of multicellular organisms. In *Proceedings of the 12th International Conference on the Synthesis and Simulation of Living Systems (ALife XII)*, pages 133–140, Odense, Denmark, 2010.

[116] Lisa Schramm, Yaochu Jin, and Bernhard Sendhoff. Redundancy creates opportunity in developmental representations. In *Proceedings of the IEEE Symposium Series on Computational Intelligence 2011 (IEEE ALIFE 2011)*, pages 203–210, Paris, France, 2011.

[117] Michael Sfakiotakis and Dimitris P. Tsakiris. Simuun: A simulation environment for undulatory locomotion. *International Journal of Modelling and Simulation*, 26(4):350–358, 2006.

[118] Shark Machine Learning Library, 2008. http://shark-project.sourceforge.net.

[119] Abdul A. Siddiqi and Simon M. Lucas. A comparison of matrix rewriting versus direct encoding for evolving neural networks. In *Proceedings of IEEE International Conference on Evolutionary Computation (CEC)*, pages 392 – 397, 1998.

[120] Karl Sims. Evolving virtual creatures. In *Proceedings SIGGRAPH*, pages 15–22, 1994.

[121] Lee Spector, Jon Klein, and Mark Feinstein. Division blocks and the open-ended evolution of development, form, and behavior. In *Proceedings of GECCO*, pages 316–323, 2007.

[122] Kenneth O. Stanley and Risto Miikkulainen. Evolving neural networks through augmenting topologies. *Evolutionary Computation*, 10(2):99–127, 2002.

[123] Kenneth O. Stanley and Risto Miikkulainen. A taxonomy for artificial embryogeny. *Artificial Life*, 9(2):93–130, 2003.

[124] Kenneth O. Stanley, David B. D'Ambrosio, and Jason Gauci. A hypercube-based encoding for evolving large-scale neural networks. *Artificial Life*, 15(2):185–212, 2009.

[125] Till Steiner, Markus Olhofer, and Bernhard Sendhoff. Towards shape and structure optimization with evolutionary development. In *Proceedings of the 10th International Conference on the Simulation and Synthesis of Living Systems (ALife)*, pages 70–76, 2006.

[126] Till Steiner, Lisa Schramm, Yaochu Jin, and Bernhard Sendhoff. Emergence of feedback in artificial gene regulatory networks. In *Proceedings of the IEEE Conference on Evolutionary Computation (CEC)*, pages 867–874, Singapore, 2007.

[127] Till Steiner, Yaochu Jin, and Bernhard Sendhoff. A cellular model for the evolutionary development of lightweight material with an inner structure. In *GECCO'08: Proceedings of the 10th Annual Conference on Genetic and Evolutionary Computation.*, pages 851–858, 2008.

[128] Till Steiner, Yaochu Jin, Lisa Schramm, and Bernhard Sendhoff. Dynamic links and evolutionary history in simulated gene regulatory networks. In Sanjoy Das, Doina Caragea, Stephen Welch, and William H. Hsu, editors, *Handbook of Research on Computational Methodologies in Gene Regulatory Networks*, pages 498–522. IGI Publishing, 2009.

[129] Till Steiner, Jens Trommler, Martin Brenn, Yaochu Jin, and Bernhard Sendhoff. Global shape with morphogen gradients and motile polarized cells. In *Proceedings of the 2009 Congress on Evolutionary Computation*, pages 2225–2232, 2009.

[130] Felix Streichert, Christian Spieth, Holger Ulmer, and Andreas Zell1. Evolving the ability of limited growth and self-repair for artificial embryos. In *Advances in Artificial Life: 7th European Conference on Artificial Life (ECAL 03)*, pages 289–298, Dortmund, Germany, 2003.

[131] Tim Taylor and Colm Massey. Recent developments in the evolution of morphologies and controllers for physically simulated creatures. *Artificial Life*, 7(1):77–87, 2001.

[132] Alan M. Turing. On Computable Numbers, with an Application to the Entscheidungsproblem. *Proceedings of the London Mathematical Society*, s2-42(1):230–265, 1937.

[133] Alan M. Turing. On Computable Numbers, with an Application to the Entscheidungsproblem. A Correction. *Proceedings of the London Mathematical Society*, s2-43(1):544–546, 1938.

[134] Erik van Nimwegen, James P. Crutchfield, and Martijn Huynen. Neutral evolution of mutational robustness. *PNAS*, pages 9716–9720, 1999.

[135] Bart W. Verdaasdonk, Bart F. J. M. Koopman, and Frans C.T. Van Der Helm. Energy efficient and robust rhythmic limb movement by central pattern generators. *Neural Networks*, 19(4):388–400, 2006.

[136] Kyriakos Voutsas. *Biologically inspired sound processing using spiking neural networks*. PhD thesis, Technische Universität Darmstadt, Germany, 2007.

[137] Andreas Wagner. Distributed robustness versus redundancy as causes of mutational robustness. *BioEssays*, 27:176–188, 2005.

[138] Duncan J. Watts and Steven H. Strogatz. Collective dynamics of 'small-world' networks. *Nature*, 393(6684):440–442, 1998.

[139] James M. Whitacre. Degeneracy: a link between evolvability, robustness and complexity in biological systems. *BMC Journal of Theoretical Biology and Medical Modeling*, 2010. Accepted.

[140] Stewart W. Wilson. The animat path to AI. In *From Animals to Animats: Proceedings of the First International Conference on the Simulation of Adaptive Behavior*, pages 15–21, 1991.

[141] Lewis Wolpert. *Principles of Development*. Oxford University Press, 2004.

[142] Alison Woollard. Gene duplications and genetic redundancy in *c.elegans*. In *Wormbook*. The C. elegans Research Community, 2005. doi: doi/10.1895/wormbook.1.2.1. http://www.wormbook.org.

[143] Tina Yu and Julian F. Miller. Through the interaction of neutral and adaptive mutations, evolutionary search finds a way. *Artificial Life*, 12(4):525–551, 2006.

[144] Song Zhan, Julian F. Miller, and Andy M. Tyrrell. An evolutionary system using development and artificial genetic regulatory networks for electronic circuit design. *Biosystems*, 98(3):176–192, 2009.

Curriculum Vitae

Personal Details

Name:	Lisa Schramm
Date of birth:	23.03.1983
Place of birth:	Frankfurt, Germany

Education and Work Experience

Since April 2011	Development engineer at Adam Opel AG
Nov. 2007 – March 2011	PhD student at control theory and robotics lab, Technische Universität Darmstadt, Germany in close cooperation with Honda Research Institute Europe GmbH. Topic: Evolution of Cellular Systems – Optimization of Gene Regulatory Networks for the Development of Morphology and Control
2007	Diploma in electrical engineering at Technische Universität Darmstadt, Germany
2002	Baccalaureate at Freiherr-vom-Stein-Schule, Frankfurt, Germany